UNIVERSITY OF CALIFORNIA

Los Angeles

Computational Methods and Models in Macroeconomics

A dissertation submitted in partial satisfaction

of the requirements for the degree

Doctor of Philosophy in Economics

by

Christopher John Surro

2020

ABSTRACT OF THE DISSERTATION

Computational Methods and Models in Macroeconomics

by

Christopher John Surro
Doctor of Philosophy in Economics
University of California, Los Angeles, 2020
Professor Pierre-Olivier Weill, Co-Chair
Professor Pablo Fajgelbaum, Co-Chair

New developments in computational power and methods have introduced a wide variety of new tools for economic analysis. This dissertation explores whether these new advances can help us understand and analyze macroeconomic phenomena.

Chapter 1 shows how recently developed clustering methods can be helpful in identifying consumer groups based on a history of purchasing behavior. While traditional methods of customer segmentation rely on observable characteristics of consumers or the products they buy, the methods I use in this chapter rely instead on identifying groups of consumers who buy a similar set of products. If consumers who buy similar products are likely to have similar preferences, then clustering groups of consumers who buy similar products can potentially uncover groups of consumers who share unobservable characteristics that drive their preference structures without explicitly specifying those preference structures. Using simulations of a discrete choice logit demand system, I show that a density peaks clustering method can effectively uncover consumers with different preferences, using a series of examples to show how different assumptions impact the effectiveness of the clustering algorithm.

In chapter 2, I apply the methods of chapter 1 to attempt to improve measurements of the cost of living. In particular, I show that methods for measuring cost of living that rely on

aggregate CES representative agents will often overstate the gains from new product varieties when groups of consumers have different tastes for products. Since the purchase data of consumers shows that there is substantial heterogeneity in the sets of products consumers buy, estimating inflation using a representative agent approach could produce biased estimates. However, the methods from chapter 1 can help to mitigate the bias from heterogeneity by grouping similar consumers based on their purchase history. I apply the method to a large panel of consumers and show that clustering consumers reduces the welfare impact of new product entry. Estimation on clustered data finds lower elasticities of product substitution and implies inflation rates about half a percentage point higher than a representative agent approach.

Chapter 3 offers a different connection between computational advances and macroeconomics by studying how visual tools can help to understand macroeconomic dynamics. I develop a visual, interactive model that allows the user to adjust parameters and observe the dynamics of the economy in real time. The model is agent based and assumes that agents make decisions based on defined heuristics rather than maximizing behavior. The core of the model draws inspiration from two sources. First, it models firm pricing behavior based on customer market models where firms attract customers by lowering prices and increase prices to earn profit on a larger customer base. Second, it assumes that firms quantity choices are determined largely by the level of demand they currently face and make adjustments on the basis of Keynesian inventory adjustments. A benchmark calibration of the model shows that the economy experiences endogenous cycles in key economic variables. Simple policy experiments show that the model responds to typical Keynesian fiscal and monetary policies.

The dissertation of Christopher John Surro is approved.

Francois Geerolf

Stavros Panageas

Pierre-Olivier Weill, Committee Co-Chair

Pablo Fajgelbaum, Committee Co-Chair

University of California, Los Angeles

2020

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

VITA

2011-2014	B.A. Economics, University of Massachusetts, Amherst
2015	M.A. Economics, UCLA
2015-2018	Teaching Assistant, UCLA
2017-2020	Instructor, UCLA
2017-2018	Teaching Assistant Consultant, UCLA

CHAPTER 1

Clustering Consumer Data: Using Density Peaks Clustering to Uncover Preference Similarity

1.1 Introduction

Even given the most detailed data on consumers' observable characteristics, an economist would have a difficult time trying to predict purchasing behavior. Consumers' preferences are diverse and idiosyncratic. Despite this reality, there is a lot of value in trying to explain consumer choice using economic data. When using a model to help understand consumer choice data, an economist will frequently need to aggregate the data, a process that usually works best when the consumers are similar. But what does it mean for consumers to be similar? This paper looks at the ability of a clustering algorithm similar to the one introduced by Rodriguez and Laio (2014) to pull out similar consumers from a panel of consumer purchase data. Rather than focus on grouping consumers based on observable characteristics, the method uses only each consumer's purchase history and groups consumers based on what they bought in the past. The method relies on the fundamental observation that consumers who have similar purchase histories are more likely to have similar preferences. Using simulations from a discrete choice logit demand system, I show that the method can often uncover groups of consumers who share similar tastes.

Customer segmentation and classification can be useful in a variety of contexts. Practically, firms are often interested in targeted advertising to groups of consumers most likely to buy their products, or organize their stores around consumers buying habits. For more theoretical purposes, segmentation can help solve aggregation problems in estimating elasticities or choice probabilities. In standard discrete choice models, aggregate choice probabilities and elasticities are generally not equal to the average of individual group probabilities when those groups have different characteristics driving their choices. However, when consumers can be accurately segmented into groups that share the same characteristics, the researcher can recover aggregate outcome variables (Train, 2009).

Typically, methods that attempt to segment consumers rely on observable characteristics. For example, given available data, a researcher might decide to group individuals by their ages, their incomes, or where they live. While these kinds of variables likely reveal some information about a consumer's underlying tastes, they largely fail to explain consumer choice (Nevo, 2011). As a result, an empirical model that relied only on observable characteristics would likely suffer many of the same issues as one that simply aggregated everyone together. One way to deal with the unobserved heterogeneity that influences much of consumer choice is to try to find the distribution of unobserved characteristics that will best allow the model to explain the observed data. This approach forms the basis for mixed logit (random coefficient) models (Berry et al., 1995). However, these models generally require relatively detailed data on product characteristics in addition to information about the consumers.

In contrast, the clustering algorithm I apply in this paper does not require any data on consumer or product characteristics because it does not try to estimate a specific form for the preferences of each consumer at all. Instead, it makes the assumption that consumers who have bought a similar bundle of products in the past probably have similar preferences. This flexibility does come at a cost. The algorithmic clustering method used in this paper makes no attempt to *explain* consumer choice. If an economist needs to know *why* certain groups of people make a certain choice, this method will be of no help. However, in many cases consumer choices simply cannot be explained by any observable characteristic. Two consumers might look identical in any observable category, but have entirely different preferences. In these cases, we can only see the differences between consumers by looking at their choice data.

The algorithm used in this paper is an adaptation of the Density Peaks Clustering (DPC) algorithm introduced in Rodriguez and Laio (2014). The DPC algorithm has been used in a variety of contexts ranging from photo recognition to gene classification (Mehmood et al., 2017). However, to the best of my knowledge, it has not been taken to economic data. I modify the DPC algorithm with some of the suggestions of Floros et al. (2018) for dealing with situations when data is large and sparse. The basic idea of the algorithm is to identify data points with many close neighbors (in this case consumers who make similar purchases to many others) and label these "density peaks." Every point whose "nearest neighbor" (i.e. the most similar consumer) is a density peak is then placed in a cluster together with the peak. The nearest neighbors of these points are placed in the same cluster until all points have been placed in a cluster. The algorithm does not require assumptions on either the number of clusters or the density of the clusters. This flexibility allows it to uncover similarity even when consumer preferences follow unusual distributions.

To illustrate the effectiveness of the clustering algorithm, I simulate data with tastes drawn from a variety of different distributions. The nature of machine learning clustering algorithms makes it difficult to prove how well the DPC process can do in clustering consumer data. However, simulations suggest that if tastes are relatively stable over time and the period of observation is long enough, then the algorithm can usually identify the correct groups. The algorithm does not do well when tastes are close to continuously distributed across consumers.

1.2 The Basic Clustering Problem

Assume that we have a group of consumers that make choices according to a standard discrete choice model. Each consumer makes a series of choices over a constant set of available products. Each time a consumer, n, makes a purchase decision, they choose the product among J alternatives that gives them the highest utility. Utility for each product can be written in the form

$$U_{nj} = V_{nj} + \varepsilon_{nj},$$

where ε_{nj} is a random term with an iid extreme value distribution.

Previous work has spent substantial time deriving the properties of this logit model (Anderson et al., 1992). For this paper, we will make some assumptions to isolate the central question. In particular, we will assume that we can write the function V as a function of price p and some unobserved factors that will be summarized by the variable ϕ. We can think of both ϕ and ε as representing factors that drive an individual's choice of good other than the price they pay, but we will assume ϕ is constant over all of the choices that a consumer makes. In other words, for every purchase they make, a consumer will draw a new random value of ε but will always have the same value of ϕ. We will assume that utility is increasing in ϕ and decreasing in price. If consumers all had the same value of ϕ (or if ϕ were observed and could be controlled for, there would be no issue in applying the standard results of logit models to calculate choice probabilities and elasticity of substitution across products. However, if ϕ differs across consumers and is unobserved, there is no guarantee that aggregate data will accurately represent the distribution across consumers. Therefore our goal will be to try to remove heterogeneity in ϕ by clustering consumers. We can then recover choice probabilities within each cluster and aggregate to the population by taking a weighted average over the clustered probabilities (as discussed in chapter 2 of Train (2009)).

To give a concrete example, we can imagine that each day a consumer goes to the grocery store to buy one item. They have some goods that they naturally like (high ϕ) and some they dislike (low ϕ). For example we could imagine a vegetarian who has a high value of ϕ for vegetables and a low value for meat products. On top of these persistent tastes, they might also have changes in their preferences each day. Maybe they are hungry one day when they enter or they just have a particular craving for one specific good. These kinds of changes will be represented by ε and assumed to be entirely random. Finally, the consumer observes prices and makes a decision about which kind of good will best satisfy their preferences and purchases that item. We can then think of creating a matrix with each consumer as a row and each product as a column. Every time a consumer buys a product, we add 1 to their row in the corresponding product column. Although in this example I frame the problem as a daily purchase, we could also imagine it as a series of purchases in a single trip to the store, with the restriction that they draw new values of ε for every product after every purchase.

Our goal in this clustering exercise will be use purchase data to group consumers who share similar values of ϕ. Returning to the example above, any vegetarian's purchase history would show that they never purchase meat even when price is low. If we attempted to aggregate this vegetarian group together with a group of meat eaters, we might get misleading results for choice probabilities. However, clustering on purchase data would allow us to separate vegetarians from consumers who frequently purchase meat. Note that the method does not attempt to explain *why* consumers would prefer one good over another. It could be that high income consumers typically buy a different set of goods than lower income consumers. It could be that consumers in different regions have different sets of goods available to them. Or it could simply be pure taste differences across consumers. Any of these reasons could explain different values of ϕ, but the method makes no attempt to differentiate between these explanations. All it aims to say is that *for some reason* this group of consumers seems to make similar purchases over time and that that pattern of purchases is significantly different from some other group of consumers who consistently purchase some other set of goods.

1.3 The Density Peaks Clustering Algorithm

Data scientists have developed a wide variety of clustering algorithms to uncover patterns in data. To choose one that suits the purpose of clustering consumer data, we need to find one that satisfies a few different properties. First, because it is usually difficult to have a prior on the number of groups in the data, we would like an algorithm that can determine the number of groups given the data. We would also like clusters of consumer data to have arbitrary sizes and shapes - some clusters might be quite large while others relatively small. These requirements make the popular k-means algorithm an ill fit for our purposes. Density based methods like DBSCAN are better able to deal with clusters with non-uniform shapes and sizes, but often require a fixed cutoff for the minimum density of each cluster. When clusters have different densities, these methods don't work well.

Recently, a new class of clustering methods has formed around the "Density Peaks Clustering" (DPC) method introduced in Rodriguez and Laio (2014), who show that it performs well in facial recognition. The basic principle of the DPC algorithm assumes that cluster centers will be both in relatively dense areas of the characteristic space and far from other points of higher density. The algorithm, then takes in points (which can represent their values along a number of dimensions) and attempts to group these into clusters that share similarity. We can define the "density" of a point in various ways. In the original DPC paper, the authors use the definition that a point's local density ρ_i is given by

$$\rho_i = \sum_j \chi(d_{ij} - d_c),$$

where d_{ij} is a measure of distance between points i and j (e.g. Euclidean distance), d_c is a chosen cutoff distance, and $\chi(x)$ is an indicator function that equals 1 if $x > 0$ and 0 otherwise. In words, the density of point i is the number of other points that are within distance d_c of i. Again, following Rodriguez and Laio (2014), we could then define another measure to represent the distance to other higher density points

$$\delta_i = \min_{j:\rho_j > \rho_i} (d_{ij}).$$

A density peak is a point where density (ρ) and distance to nearest point of higher density (δ) are both high. These points have many close neighbors, but are far away from other cluster centers. Once density peaks have been identified, remaining points are allocated to the nearest density peak to form a cluster. Points that have high δ but low ρ are discarded as outliers because they are points that are not close to any of the densely populated clusters.

As a relatively simple and efficient mechanism for clustering data, the DPC algorithm offers a promising way to find heterogeneity in consumer tastes. It only relies on calculating distances between points and choosing a single parameter d_c. Unlike more standard estimation methods, it does not require specifying a probability distribution or a functional form. It allows for flexibility in both the number and shape of clusters and can be implemented in a computationally efficient way. However, a few concerns remain.

One potential issue with the DPC algorithm in its basic form is that it can be difficult to set the cutoff point d_c, especially when data is high dimensional. Since in my application, consumers could potentially be consuming many different products, the characteristic space can be high dimensional and distances between points can be quite large. Problems with high dimensional data can sometimes be solved by using relative measures of distance rather than absolute. K-nearest neighbor methods offer a way to accomplish this goal. Ertöz et al. (2003) argue that some of the issues of high dimension can be solved using a nearest neighbor approach instead of methods that rely on absolute distances.

To adapt the definition of density to make it a relative measure based on nearest neighbors, I use some of the adjustments to the algorithm presented in Floros et al. (2018). I first find the k nearest neighbors of each point, where k is a parameter that can be freely chosen and define density as

$$\rho_i = 1/\max_j (d_{ij} | x_j \in \mathcal{N}_k(x_i)),$$

where $\mathcal{N}_k(x_i)$ is the set of the k nearest neighbors of point i and d_{ij} is again any distance measure between points. In words, this measure defines density as the distance from each point to its k'th nearest neighbor. If a point is relatively close to all points in its neighborhood, it is considered dense. Compared to the original distance measure defined above, this definition is more suited to high dimensional data where defining a specific cutoff point can be challenging. Choosing too low a cutoff will result in all points having density zero. Too high, and all points will be in the same radius. Here, density is a relative measure that compares the distance to the furthest neighbor among a small set of the closest points.

Again, following Floros et al. (2018), I make the density peaks local maxima of their neighborhood. In other words, if a point i has higher density than any of its k nearest neighbors, it will be labeled a density peak. Compared to standard DPC, this definition is computationally more feasible since it only requires distances to be calculated between nearest neighbors rather than the entire set. To assign non-peak points to clusters, we will again use the same definition of δ to signify distance to higher density points

$$\delta_i = \min_{j:\rho_j > \rho_i} (d_{ij}).$$

From there, we will define a point's *parent* as the nearest neighbor of higher density

$$x_{ip} = \arg\min_j (d_{ij} | \rho_j > \rho_i).$$

By definition, any density peak will not have a parent in their kNN list since they are the point of highest density among their nearest neighbors. Every other point will have a parent within their set of k nearest neighbors. Linking each point to its parent node will form a path that eventually reaches a density peak. Each point in this ascending tree is then placed into the same cluster as the density peak at its end. Because the method is still somewhat sensitive to choices of distance measure, noise in data, and tie-breaking in selection of parents, it is usually necessary to revise the clusters after the initial allocation. I discuss the revision process in more detail in the appendix.

DPC has not yet been applied to a dataset of consumer purchases, but the methods carry over quite naturally. The set of characteristics in this case corresponds to the set of products available to consumers. To avoid clustering only on the amount of purchases a consumer makes, we will look at the share of each product in a consumer's total expenditure. For example, if we had two consumers and four products, we might have a matrix of shares that looks like

$$\begin{bmatrix} 0.1 & 0.4 & 0.4 & 0.1 \\ 0.25 & 0.25 & 0.25 & 0.25 \end{bmatrix},$$

where the rows represent consumers and the columns represent products. We could then take the Euclidean distance between these two vectors (in this case about 5.4) to quantify their differences. We can then apply the method above. The k closest consumers to consumer i as measured by Euclidean distance are placed in the nearest neighbor set of consumer i and the distance to the furthest point in that neighborhood is the density. Figure 1.1 shows an example for two small clusters of consumers who purchase 4 products. The plot on the left shows the share data for goods 1 and 2. We can see that some consumers consume a large portion of good 1 and not so much good 2, while another group seems to do the opposite. The graph on the right plots density δ against distance to nearest higher density point δ. We can see that there are two outliers on this graph (highlighted in black) that correspond to the density peaks. They are close to all the points within their respective clusters while being far away from each other (the next highest density point).

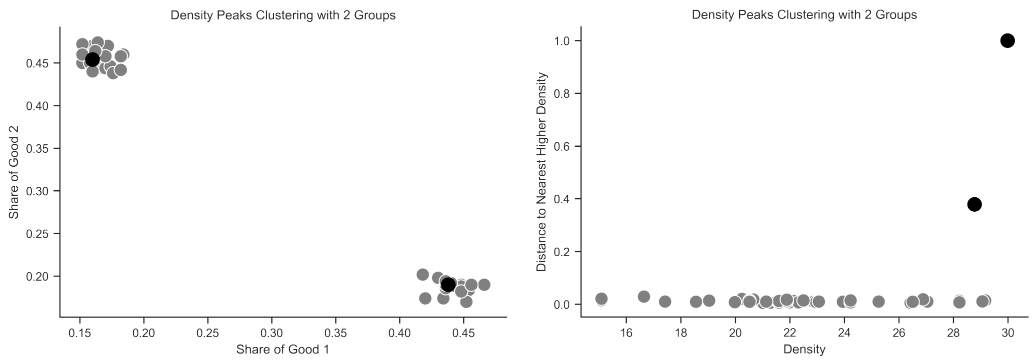

Figure 1.1: An illustration of the Density Peaks Clustering algorithm with two groups. Left: Sample share data for consumers buying 4 goods. The graph shows their shares for two of these goods. Right: Density against distance to next point of higher density. The two outliers highlighted in black are the density peaks.

While the clusters in this basic example are quite easy to spot, they will not be so evident when we have many products and many goods. In the next section, I show through simulations that the algorithm can accurately uncover groups of consumers with different tastes even in these more complicated cases.

1.4 Simulations

As mentioned earlier, algorithmic clustering methods do not lend themselves to rigorous proofs of their effectiveness in identifying groups. Instead, this section provides a variety of simulations to show how the algorithm responds to changes in the underlying assumptions of the demand system. To make terms clear, note that I will always use *group* to refer to the true groups of consumers with similar preferences and *cluster* to refer to the clusters that the algorithm generates.

1.4.1 Illustrative Example

We will start by looking at an example where we would expect the algorithm to perform well in order to have a point of comparison when we use more challenging situations. The goal in this section is to show that the method works under ideal conditions and to demonstrate what it looks like when it works well. As described above, we will assume consumers make choices based on

$$U_{it} = \ln(\phi_k^i) - \ln p_{tk} + \varepsilon_{itk}.$$

For the simulations, we will assume that prices are drawn i.i.d from a lognormal distribution each period and ε is drawn from a Gumbel distribution with location parameter equal to zero and shape parameter equal to 1/5. Each period, consumers will make a single choice out of 20 total products. Each group has a slightly stronger taste for two out of the twenty products (with no overlap) and there are 10 groups total with 1000 consumers per group. Each consumer has a taste $\phi = 2$ for their preferred goods and $\phi = 1$ for all others. The simulation runs for 100 periods so each consumer makes 100 purchases total. The exact numbers chosen do not substantially affect the results. The important point for now is that we could easily identify the groups just by looking at the data. In this case, the clustering algorithm should easily be able to capture the groups.

Because of the symmetric nature of the setup, aggregate purchase shares in the simulation all hover around 1/20. However, within groups, the shares are much more heavily weighted to the two goods that group prefers. Figure 1.2 shows this effect. On the left, the aggregate shares indicate that the representative agent does not have a strong preference for any good. They consume approximately equal shares of every good. However, this uniformity hides the true heterogeneity within groups. Each of the colored bars represents shares for a group of consumers. Each group spends around half their expenditure on their favorite two goods while splitting the other half of expenditure about evenly among the other 18. This heterogeneity is what the clustering algorithm picks up. It looks for consumers who have disproportionately high shares of consumption for certain goods and places them together in a cluster. The stark differences in this example make this process quite easy. In the next section, we will relax some of the assumptions to see how well the algorithm performs when differences are not as apparent.

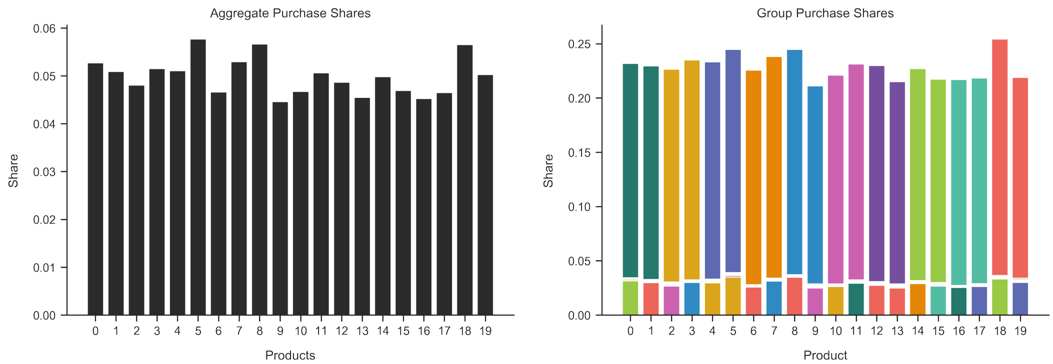

Figure 1.2: Aggregate shares (left) vs clustered shares (right). In aggregate, purchases are split evenly across the 20 goods. When the shares are plotted by cluster, it is clear that consumers all have higher purchase shares for two of the goods.

This simulation also gives an opportunity to discuss the revision process briefly mentioned above. If we run the density peaks method as described in the previous section, we will likely not recover the correct groups immediately. The number of density peaks we find will depend in most cases on the choice of k. Choosing a low value for k will create many small clusters, while a high value will result in fewer, large clusters. Rather than play around with different values for k, we can use a split and merge revision process to help refine our clusters. Since we formed our clusters by combining ascending trees (the path that connects each point to its nearest neighbor of higher density leading to a density peak), we can first look for subtrees that do not fit well in their current cluster. More specifically, we will break apart a subtree from its cluster whenever the similarity of consumers in a subcluster are much more similar to other consumers in that subcluster than they are to other consumers in the overarching cluster. We then merge clusters where consumers share a high similarity. More specifics are given in the appendix.

Figure 1.3 shows this revision process. The block partition graphs shown in the figure plot a point for every nearest neighbor pair. In other words, if consumer 500 is in the nearest neighbor list of consumer 1, a point would be plotted at (1,500). The graph is sorted by cluster so that each square along the diagonal represents a cluster. We can see that in the first round, the density peaks algorithm produces more clusters than the true 10 different groups. Since we chose a k relatively small compared to the total number of consumers in each group ($k = 50$), the algorithm is biased towards finding smaller clusters. It does a good job of avoiding mixing different groups together, but it places consumers who are actually in the same group in different clusters. the first graph on the left, many consumers have nearest neighbors outside their cluster. By merging these similar clusters, we get the graph on the right, where consumers in each cluster share many nearest neighbors within the same cluster, and very few (essentially zero) outside their cluster.

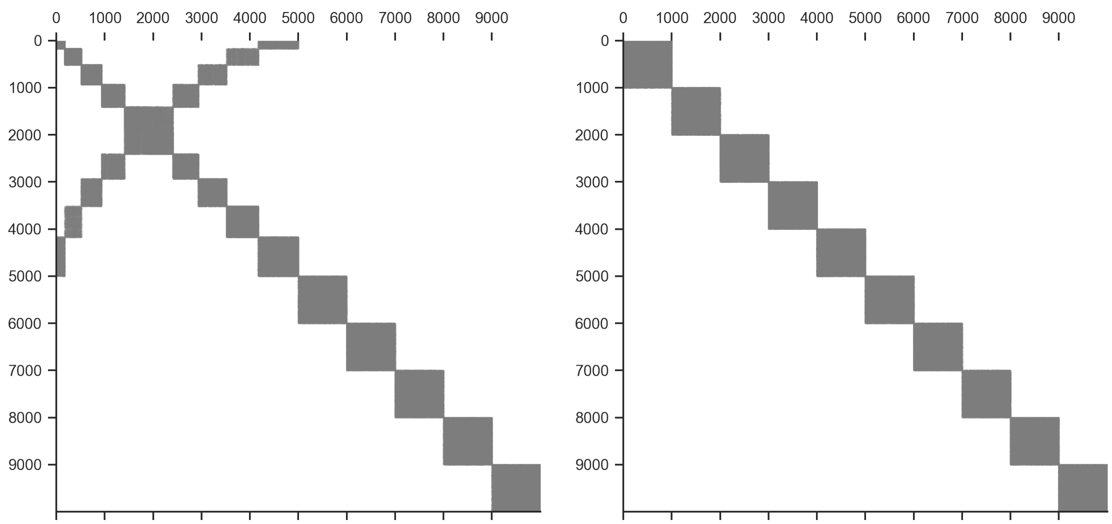

Figure 1.3: Clustering results before (left) and after (right) the revision process. There are 10 true groups of 1000 consumers each. Clusters are plotted along the diagonal and each point on the graph is a nearest neighbor pair.

To make the evaluation of clustering more concrete, we can compute some statistics to help quantify how well the algorithm does. First, we can look at the number of clusters generated compared to the actual number of groups. In the example above, before the revision, the algorithm generated 15 clusters and we have 10 true groups. We will define *clusters/group* as the number of clusters divided by the number of groups (in this case 150%). Next, we will define *cluster purity* as the percentage of points in each cluster that belong to the same group. For example, imagine the matrix below represents a clustering outcome from 2 groups. The rows of the matrix represent each cluster and the columns represent the groups

$$\begin{bmatrix} 8 & 1 \\ 2 & 9 \end{bmatrix}.$$

In words, if we call the clusters 1 and 2 and the groups A and B, this matrix means cluster 1 contains 8 consumers from group A and 1 from group B and cluster 2 contains 2 consumers from group A and 9 from group B. We calculate the purity by taking the proportion of consumers who are placed in a cluster with consumers from their own group. In the first cluster 8 out of 9 consumers come from group A and in the second group, 9 out of 11 come from group B. Taking a weighted average of these individual values, we get a purity value of 17/20 or 85%. More simply, we can calculate purity as the sum of the maximum of each row divided by the total number of consumers.

While purity gives a good measure of success when the number of clusters is equal to or less than the number of groups, it can give misleading results when there are more clusters than groups. For example, if our clusters are

$$\begin{bmatrix} 3 & 0 \\ 7 & 0 \\ 0 & 2 \\ 0 & 8 \end{bmatrix},$$

then the purity would be 100% because no consumer is placed in a cluster with a consumer from another group. However, consumers from the same group are sometimes placed in different clusters. Therefore, we can define *cluster recovery* as the percentage of a group that is placed into a single cluster. In this case, 7 out of 10 from group A are placed in cluster 2 and 8 out of 10 of group B are placed in cluster 4 so we would calculate the cluster recovery as 15/20. Here, we take sum of the maximums of each *column* divided by the total number of consumers.

These three statistics are certainly not the only way to evaluate the success of the clustering algorithm, but they provide a good picture of the relative ability to recover the groups accurately. In the simple example above, before revision we have a cluster amount at 150%, cluster purity at 100%, and cluster recovery around 80%. After revision, the clusters produced by the algorithm exactly match the original clusters.

In the next sections, we will look at a series of examples to show how well the clustering performs as we move away from the ideal assumptions given in this section. In particular, we will look at how well it holds up as we change the variation in tastes across consumers, the number and size of underlying groups, price dispersion, and the number of purchases a consumer makes.

1.4.2 Taste Variance

Intuitively, we would expect the clustering algorithm to work well only when there exist meaningful differences in consumer taste for different goods. To examine how sensitive the algorithm is to this effect, we can change the variance of ϕ across consumer groups. I stick to the function form

$$U_{it} = \ln(\phi_k^i) - \ln p_{tk} + \varepsilon_{itk},$$

but rather than assume that each consumer has a favorite good for which ϕ is higher, I will assume that each group of consumers draws their ϕ for each from an independent uniform distribution (once drawn, this value is constant over time). I normalize the lower bound of the distribution to 1 so that by varying the upper bound I can change how much tastes differ across consumers. Here I will use a simulation of 10 groups of 1000 consumers making choices across 50 different products for 50 periods. As before, within each group, consumers share a ϕ, but now these parameters are drawn randomly from a uniform distribution. Prices and idiosyncratic ε shocks follow the same distributions as in the previous section.

Figure 1.4 demonstrates the difficulty in finding the different groups when we deviate from the structure of the illustrative example. Even a relative minor change in the setup of the problem makes it much more difficult to visualize the correct groups. Unlike in the previous example, some groups have similar preferences for individual goods and simply observing which goods each consumer buys most frequently would be insufficient to reveal which group they belong to. However, the DPC algorithm is still able to find patterns that we would not be able to see just by observing the data.

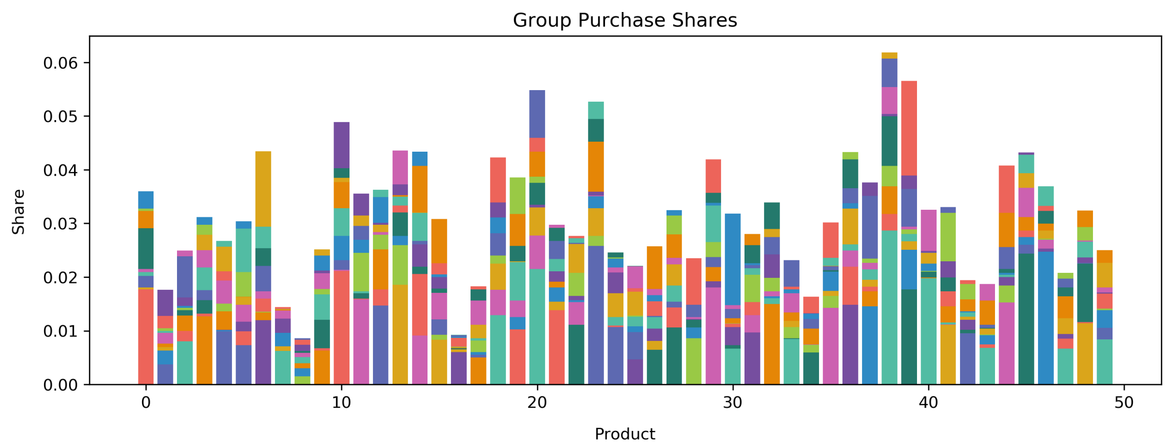

Figure 1.4: Group purchase shares with uniformly distributed tastes. Purchases of 10 groups over 50 goods and 50 periods

Figure 1.5 shows how well the algorithm does in recovering the correct groups under various levels of the variance of the distribution of potential tastes. From the figure, it is obvious that the algorithm cannot effectively discover the groups when the variance is very low. Unsurprisingly, if there is not much difference in taste across consumers, the randomness in prices and ε shocks overwhelms any effect of having a common ϕ, making it much more difficult for the algorithm to find the correct groups. However, as the variance increases to even a marginally higher value, differences between groups become more pronounced and the algorithm accurately classifies consumers into the correct groups.

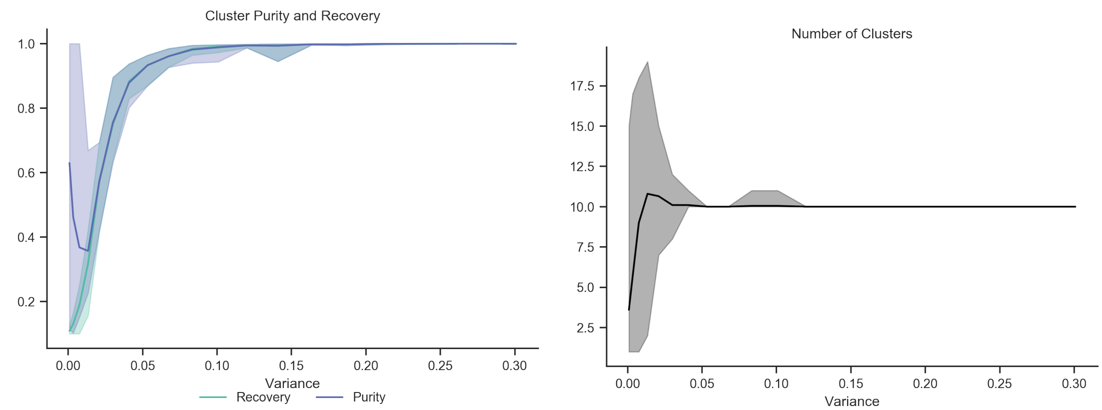

Figure 1.5: Clustering results for different levels of variance in the distribution of potential tastes. As variance increases, recovery and purity (left) increase to 100% and the number of clusters (right) correctly picks up the 10 groups. Simulation is repeated 20 times for each level of variance. Error bars show the minimum and maximum values of these simulations.

1.4.3 Arbitrary Size and Number of Groups

One of the advantages of the Density Peaks Clustering Algorithm over alternative methods is that the number and size of groups does not need to be specified by the researcher. In this section, I run a number of simulations with random numbers of clusters of arbitrary size. The simulations use the same structure as the previous section with tastes drawn from a uniform distribution with taste range from 1 to 2. Groups are generated randomly from a set of 20000 consumers and are restricted to be between 500 and 2000 consumers. The table below describes the results from 1000 random simulations given this setup. The number of groups varies from 12 to 22 and the algorithm recovers the exact number of groups in most cases (about 80%). Even when the number of clusters ends up being slightly different than the number of groups, purity and recovery rates are strong, meaning that the algorithm is doing a good job putting consumers with the same preferences in the same cluster.

Table 1.1: Summary Statistics for DPC Algorithm on Randomly Sized Clusters (1000 simulations)

	Minimum	Maximum	Median
Number of Groups	12	22	17
Number of Clusters	12	22	16
Clusters/Groups	0.79	1.07	1
Cluster Purity	0.92	0.99	0.97
Cluster Recovery	0.80	0.99	0.97

Changing the number and size of the consumer groups presents a couple of challenges for the algorithm. First, as the number of groups grows large, it becomes increasingly likely that two groups will be too similar for the algorithm to notice the differences. In other words, if the distribution of tastes across consumers gets closer to being a continuous distribution, the algorithm will naturally have trouble finding discrete groups (since discrete groups really don't exist at all in this case). On the other hand, if groups are very large (and the number of periods comparatively small), it becomes more likely that the algorithm mistakenly splits clusters. For the groups in these simulations, the algorithm does pretty well in recovering the groups. If we had used a lower bound for the minimum group size, the algorithm tends to be overly aggressive in clustering (so we get too few clusters relative to the true number of groups), and if we increase the maximum group size too much, the algorithm tends to split clusters too much. Additionally, with different numbers of consumers in each group, tuning parameters for the sensitivity of the revision process becomes difficult. The appendix discusses these tradeoffs in more detail.

1.4.4 Price Dispersion and Logit Shock Variance

One of the main issues that can prevent the DPC algorithm from accurately clustering consumers is when there is large variation in either the prices of available goods or in the idiosyncratic ϵ shocks. To see why, note that given the functional form used, there will always be some price high enough to prevent a consumer from buying a product they really like or low enough to induce them to buy a product they are not especially fond of. Therefore, if prices are more dispersed, it is more likely for consumers to gravitate towards products that have lower prices whereas when prices are similar, consumers will buy the product they have higher taste for. Similarly, if the extreme value distribution for the ε shocks has a fatter tail, it becomes more likely that consumers will draw large ε shocks that overwhelm any differences in ϕ across consumers. Intuitively, both of these sources add noise to the decision-making process that hides the source of purchase variation we are actually interested in uncovering.

Figure 1.6 below shows the results of simulations with different amounts of price variation. Recall that prices in the simulations so far have been randomly drawn from a log-normal distribution. In this exercise, I change the variance of that distribution to adjust the dispersion in prices. As expected, as the variance increases, the precision of the clustering decreases. Changing the variance of the ϵ shocks produces a similar result (not shown here).

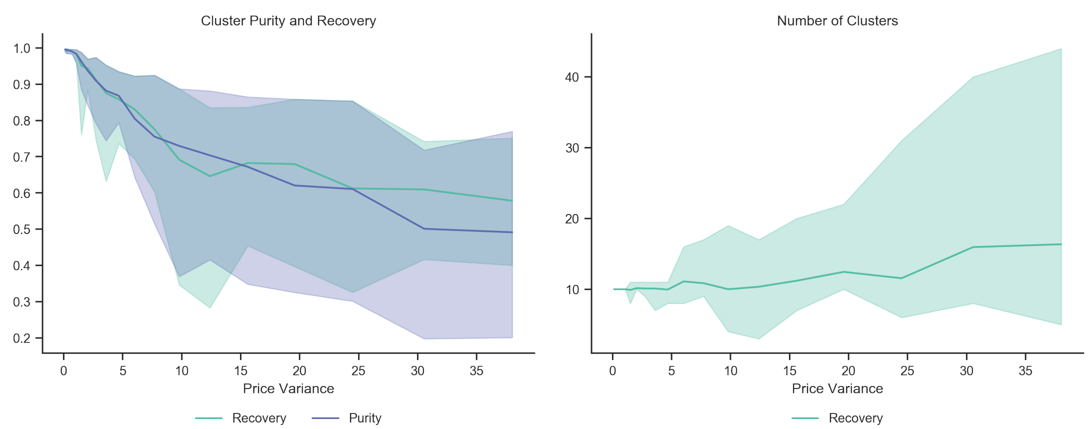

Figure 1.6: Clustering results for different levels of variance in prices. Each simulation is over 50 periods, with 50 goods being purchased by 10 groups of 1000 consumers each. As dispersion in prices increases, recovery and purity (left) become much less precise. The number of clusters also fails to match the true number of groups. Simulation is repeated 20 times for each level of price dispersion. Error bars show the minimum and maximum values of these simulations.

It is important to note that as long as we are dealing with a relatively homogeneous category of goods, the high levels of variance shown here would almost certainly be much larger than price differences observed in reality. Even a variance of 5 here implies the highest price product will be about 100 times more expensive on average than the cheapest one. This issue can largely be avoided by using a set of products that don't vary too much in price.

1.4.5 Length of Simulation

In most cases, the success of the algorithm can always be improved by increasing the number of periods (i.e. the number of purchases each consumer makes). Since we assume that all consumers face the same prices and ε shocks are i.i.d, differences in ϕ across consumers will be easier to pick up the more data the algorithm has to work with. For example, as we saw before, when the taste differences between consumers are small, the algorithm doesn't do very well in recovering the correct groups. However, if we extend the simulation long enough, it eventually does correctly classify consumers. Using the lowest range from that example, it would take around 2000 periods to get close to 100% purity and the correct number of clusters. In all of the changes discussed above, the algorithm uniformly does better as the number of periods in the simulation increases.

1.5 Conclusion

As machine learning continues to advance, economics will have much to gain from exploring algorithmic methods. This paper explored one potential avenue by using clustering techniques on consumer purchase data. Using this kind of method can enable classification of consumer groups based on unobserved heterogeneity in tastes without needing to directly specify how those tastes are distributed across consumers. In applications where the presence of unobserved heterogeneity could present problems for estimation, the density peaks clustering algorithm could help to remove that heterogeneity, allowing for estimation on more homogeneous groups. Chapter 2 of this dissertation demonstrates how this method can be used to help estimate an aggregate price index when there is consumer heterogeneity.

1.6 Appendix: Implementation of the Density Peaks Clustering Algorithm

The goal of this appendix is to provide complete details on how to implement the Density Peaks Clustering algorithm on consumer data. As part of the appendix, I will discuss some of the tradeoffs in tuning the parameters of the algorithm and how they should be changed to reflect the goals of the researcher.

1.6.1 Preparing the Data

The algorithm works with a matrix of data that describes the share of each product in a consumer's purchase history. The rows of the matrix represent each consumer and the columns represent each product. If a consumer has not ever purchased a product, the corresponding column should be a zero. Depending on the goals of the researcher, shares can represent either a share of expenditure in dollar values or quantity shares.

If consumers are exactly identical in their purchase histories, we will always assume that they are in the same cluster. If a consumer has more duplicates than the number of nearest neighbor distances calculated (more on this below), the algorithm will never include other consumers in the same cluster. To avoid this issue, I drop all duplicates before running and then add them back into their respective clusters in the final step. When the number of products and periods is large, duplicates are rare and we do not need to worry about this distinction.

1.6.2 Calculating Pairwise Distances

Ideally, to find how similar consumers' purchase histories are, we could calculate a distance measure between each pair of consumers. In practice, this calculation quickly becomes unfeasible as the number of consumers grows. In a dataset with N consumers, calculating and storing pairwise distances requires an $N \times N$ distance matrix. Fortunately, the algorithm only needs to know distances between a consumer and their k nearest neighbors, meaning we only need to store $N \times k$ actual values.

The question then becomes how can we calculate the nearest neighbors without calculating the entire pairwise distance matrix. To improve the speed and reduce the storage requirements of this calculation, I use a method from Bernhardsson (2018) that approximately finds the nearest neighbors in any collection of vectors and calculates the distances between these nearest neighbors. While an exact method would be preferred other things equal, using approximate methods provides a measure nearly as accurate and significantly faster.

Another important choice to make is what distance measure to use. There are a number of different distance measures that could potentially be used to compare consumer purchases (see Pandit and Gupta (2011) for a discussion of some of these different measures). For the simulations in this paper, we have stuck to the Euclidean distance measure. Because we have normalized our data to be between 0 and 1 by transforming it into shares rather than absolute purchase numbers, we do not have to worry as much about the tendency for Euclidean distance to overweight large values when calculating distance. Euclidean distance seemed to give the most intuitive orderings of similar consumers compared to other distance measures, but it may be worth exploring alternatives depending on the application.

1.6.3 Finding Density Peaks and Constructing the Initial Clusters

Once we have nearest neighbor lists for each consumer in the data, we can start to find the density peaks. As discussed above, we define the density of a consumer as

$$\rho_i = 1/\max_j(d_{ij}|x_j \in \mathcal{N}_k(x_i)),$$

where $\mathcal{N}_k(x_i)$ is the set of the k nearest neighbors of point i and d_{ij} is the Euclidean distance between points i and j. In other words, the density is one over the distance to a consumer's k'th nearest neighbor. We then compare the density of each consumer to the density of every consumer in its nearest neighbor list. If a consumer has a greater density than *any* of the other consumers in its nearest neighbor list, we label it a density peak.

By definition, every point that is not a density peak must have a point that has higher density. We call the closest point of higher density to each point its *parent*. We then create a set of *trees* as follows. Starting from a density peak, place any *children* (i.e. points whose parent is that density peak) into the same tree. For each of those children, find any additional children who have not already been placed into a tree and continue until reaching a point who is not the parent of any other node. Then move on to the next density peak. From the way we have discovered density peaks and defined parents, this procedure will necessarily place all points into exactly one tree. These trees form our initial clusters.

To help with the next steps, it is useful at this point to create a weighted K-Nearest-Neighbor (KNN) matrix showing the relationships between each point. To create this matrix, we calculate, for each of the nearest neighbors, we can calculate a value (suggested in Floros et al. (2018))

$$\exp\Big(-(d_{ij}\rho_i)^2\Big),$$

that represents how close a point is to each of its nearest neighbors. To each of the points that is not in the nearest neighbor list we assign a value of zero. We can then compare the strength of this measure within clusters vs across clusters to govern a split/merge revision process.

1.6.4 Revision Process

The revision process will be based on the idea that we want to maximize the size of our clusters while ensuring that each cluster retains a degree of similarity with all other consumers in the cluster. To accomplish this goal we rely on the tree structure of our clustering method. Recall that our initial clusters were generated by building up a tree of parents and their children. Our revision process will examine whether splitting off a subtree and moving it to another cluster would increase the sum of the values in each cluster in the KNN matrix.

For example, imagine we had two true groups but our initial clustering algorithm actually generated three clusters. For simplicity assume that cluster 1 has all group 1 consumers, cluster 2 has all group 2 consumers, and cluster 3 has half group 1 and half group 2 consumers.

What we will try to do is split the group 1 consumers and group 2 consumers from cluster 3 and merge them into clusters 1 and 2 respectively. Assuming consumers from each group end up in different subtrees, we can do this through the split and merge process.

Practically, it is computationally taxing to check every possible combination of subtrees. To implement a feasible version of the revision process, I implement the following structure. First, measure the mean value of the KNN matrix within a cluster. Next, pull out a subtree of that cluster and measure the mean KNN value within that subcluster. If the mean value is significantly greater than the full cluster mean value, we split that subtree into a new cluster. Defining what "significantly" means in this case is somewhat arbitrary. We can set a cutoff value between 0 and 1 such that multiplying the cutoff value by the split mean is still larger than the full cluster mean. For this paper, I use a split cutoff of 0.2, meaning that the split strength must be 5 times as large as the cluster as a whole.

Once we have split off these smaller clusters, we check whether it would be better to merge them with existing clusters. Here, we go the opposite direction and check if the value of the KNN matrix within a cluster would grow if we were to merge it with an existing cluster. Once again, we can set a cutoff level to govern how strong we need the difference to be to merge 2 clusters. I use the same 0.2 value as a cutoff for merges. Finally, I merge all clusters that have fewer than some set amount of clusters. In many cases, the process creates some tiny clusters. If any estimation needs to be done at a cluster level, clusters that have very few consumers will not be very helpful. One option would be to simply drop these small clusters as noise. In most of the simulations above, I instead choose to merge each cluster under 100 consumers into the cluster that would maximize the sum of the values of the within cluster KNN matrix.

1.6.5 Parameter Sensitivity

The algorithm has a few main parameters that need to be set. First, the number of nearest neighbors k needs to be chosen. Choosing a small k will tend to find many density peaks and therefore create many initial clusters. While this can sometimes be desirable because it gives the split/merge revision process more flexibility, choosing a k too small will usually result in more clusters than true groups. It increases the tendency for the algorithm to find random correlations between consumers. It also slows down computation time since the split and merge process will need to check a much larger set of initial clusters. On the other hand, choosing a k too large will increase the likelihood of underestimating the number of groups. Trivially, if k is set to equal the number of consumers, all consumers will be placed in the same initial cluster. In this case, there is nothing the revision process can do to improve the situation. Therefore, it makes sense to choose a relatively small value of k if the size of groups is unknown (as will usually be the case).

Setting parameters for the revision process also presents some challenging tradeoffs. Setting the split cutoff very high (meaning the strength of a split needs to be relatively low to break off from a cluster) will create many small clusters and increases the possibility of splitting groups into multiple clusters. Alternatively, setting the merge threshold too low will lead to clusters being merged even if they include consumers from different groups.

In most cases, it will usually be less harmful to have too many clusters than too few. In other words, it is better to split a group into 2 clusters rather than mix two groups into one cluster. However, as discussed above, having too many clusters makes estimation difficult as the number of consumers in each cluster gets small. Depending on the application, the researcher will need to decide how to balance this tradeoff.

CHAPTER 2

The Cost of Heterogeneity: Can Density Peaks Clustering Improve Estimation of CES Price Indexes?

2.1 Introduction

[1]

Coming up with some measure of the average cost of living has been a challenge for economists and policymakers since price data started to become available. Purely statistical methods for averaging prices over the entire economy quickly run into difficulty when consumers can substitute expensive goods for cheaper goods or when new goods are introduced. Empirical work has shown that these substitution and new product effects are quantitatively important (Boskin et al., 1997). More theoretical approaches that attempt to model consumer preferences can account for substitution and product entry and exit, but usually require taking restrictive stances on the underlying preferences of consumers (Feenstra, 1994). Recently, the availability of detailed micro level sales data has allowed researchers to estimate more detailed models of consumer preference, which have revealed that these substitution and new product effects can substantially change estimates of aggregate price indexes (Argente and Lee, 2019; Broda and Weinstein, 2010).

[1]Researcher(s) own analyses calculated (or derived) based in part on data from The Nielsen Company (US), LLC and marketing databases provided through the Nielsen Datasets at the Kilts Center for Marketing Data Center at The University of Chicago Booth School of Business. The conclusions drawn from the Nielsen data are those of the researcher(s) and do not reflect the views of Nielsen. Nielsen is not responsible for, had no role in, and was not involved in analyzing and preparing the results reported herein.

This paper expands on recent work that estimates CES demand systems using micro-level data by trying to account for the consumer heterogeneity apparent in the data. I show that estimating an aggregate CES preference structure when the true model has groups of consumers with different tastes will result in elasticity estimates that lie below the elasticities of each of the individual groups. To improve the estimation, I use the clustering algorithm developed in chapter 1 of this dissertation to cluster consumers based on their purchase data and then estimates demand parameters separately across each of the groups. While other papers in the macroeconomics literature have attempted to use observable demographic characteristics to partition the customer space, this paper is among the first to use clustering techniques developed in other fields to identify unobserved heterogeneity among consumers. By connecting these related but often isolated literatures, I shed new light on the impact of product creation and destruction on consumer welfare. Compared to a representative agent approach, I find lower levels of substitution, which implies a lower impact from the introduction of new varieties on welfare.

The standard method for dealing with the issue of substitutability and new products in the macroeconomics literature is to define preferences of a representative consumer and estimate parameters governing substitutability. Feenstra (1994) showed how to separate an aggregate price index derived from a CES utility into components governing continuing and new products in a trade context and similar methods have been applied in a variety of contexts since this seminal work. These methods bestow an important role on the elasticity of substitution in quantifying the impact of new products on the cost of living. If products are highly substitutable, then product entry and exit matters relatively little to consumers since products can be easily replaced by a close substitute. However, when products are highly differentiated, new products can substantially increase consumer welfare since they are more likely to satisfy some consumer's preference (and conversely exiting products can substantially decrease welfare). The Feenstra method has been applied to micro data in various recent papers that have found substantial bias in price indexes that ignore product innovation (Broda and Weinstein, 2006, 2010; Hottman et al., 2016)

Standard CES models are already equipped to deal with some level of consumer heterogeneity. Previous work has shown that a discrete choice model where consumers choose a single differentiated product can aggregate to a CES representative agent (Anderson et al., 1992). However, this justification for using CES is somewhat fragile. Recent work has shown that the logic behind CES aggregation breaks down when price changes are asymmetric (Tito, 2016). There are also issues when we try to estimate elasticities on aggregate data. When there is heterogeneity across goods, aggregated data often implies an elasticity lower than the average of microeconomic estimates (Orcutt, 1950; Imbs and Mejean, 2015).

I focus on a different dimension of heterogeneity by looking at systematic differences in consumer preferences. When consumers differ in their average tastes for goods (i.e. not only due to taste shocks independently distributed over time), I show that the elasticity implied by the aggregate data is usually below the individual estimates within the heterogeneous groups. Intuitively, if each consumer only consumes a subset of available goods, they might have high substitution rates within their preferred product group, but essentially no substitution outside of it. A representative agent mixes these two effects, which makes it appear as if it has a lower elasticity across all products. As a result, the effect of new products on the price index is amplified by the representative agent. When net product creation is positive, this result implies that the aggregate representative agent could face a larger decrease in the cost of living due to product entry than *any* of the individuals in the economy. Motivated by this observation, I conclude that to accurately represent the inflation rate for consumers in the economy, we need to deal with this heterogeneity in average tastes before estimating the demand system.

Departures from the representative agent have been explored in previous work. Osharin and Verbus (2018) show the effects of letting consumers have different elasticities on markups and wages. More closely related to this paper, (Jaravel, 2019) found evidence of inflation heterogeneity when looking at inflation rates for different income groups. However, while observable characteristics like income certainly drive some of the variation in consumers' preferences, it is unclear whether they are the most important factors. In a survey of recent methods for modeling consumer behavior, Nevo (2011) remarks that "heterogeneity in choice is only weakly correlated with standard consumer attributes. Income, education and family size obviously explain some dimensions of choice, but are far than enough to accurately predict consumer behavior. Unobserved heterogeneity is important to model in many cases" (4). How can we distinguish between groups of people with different preferences if those preferences are driven by unobservable factors?

One way to handle unobserved preference heterogeneity is to make assumptions about its underlying structure. If unobserved features are drawn from a known distribution, in many cases a researcher can uncover its parameters based on choice data. These kinds of methods underlie many of the popular discrete choice estimation procedures with random coefficients and mixed logit specifications. A line of research stemming from Berry et al. (1995) has developed tools for dealing with unobserved heterogeneity by using instrumental variable identification strategies to identify structural parameters of consumer demand for product characteristics. While these methods offer a path forward for dealing with consumer heterogeneity, finding valid instruments is especially difficult in a macro setting where classic instruments (e.g. prices of same goods in other markets) are often inappropriate. In many cases, these methods also require functional or distributional assumptions about the sources of heterogeneity (although these assumptions are beginning to be relaxed as in Compiani (2018)). Finally, CES preferences have retained their prominence in large part due to their tractability and nice aggregate properties, especially in allowing for product entry and exit. More flexible models often complicate the aggregate interpretations produced by the model.

To allow for heterogeneity while retaining the tractability and convenient properties of CES, I take an alternative approach by using the algorithm described in chapter 1 to uncover patterns in the data without taking a stand on the source of the taste heterogeneity. Applying the methods of Redding and Weinstein (2019), I show that the estimates in their work cannot be interpreted as an average across consumers in cases where preferences differ systematically. However, by first assigning these consumers to the proper clusters, my method in theory gives representative numbers *within* clusters. I apply the algorithm to consumer panel data to estimate an aggregate price index for the US economy retail sector from 2004-2017. Since the data consists of multiple purchases by the same consumers over time, the clustering algorithm is able to uncover differences in consumer buying behavior and group consumers who buy a similar basket of goods. Most product groups in the data are characterized by a few major brands that dominate product sales. The clusters uncovered roughly correspond to consumers who disproportionately purchase one brand over the others. I show evidence that the clustering algorithm effectively creates clusters of consumers who have similar patterns of expenditure and different patterns from those consumers in other clusters.

As predicted by the theory, I find higher elasticities for within group preferences than for an aggregate represenetive agent. On average I find elasticities about 20% higher within clusters than for a representative agent (7.8 and 7.1 respectively). I then estimate a price index first for a representative agent that ignores group heterogeneity and then within each of the groups uncovered by the clustering algorithm. The difference between the two indexes depends on how consumers are weighted, but benchmark results that weight groups by their expenditure show that the representative agent approach understates the inflation rate by approximately 0.5 percentage points per year over a 10 year period.

Machine learning is still relatively underused in economics, but it has the potential to make a major impact. Recent economic research has shown that it has many possible applications (Athey and Imbens, 2019). This paper will try to continue to add to that growing literature.

2.2 Product Market Stylized Facts

2.2.1 Data Description

To find some stylized facts about consumer purchasing behavior, I use the Nielsen Homescan Consumer Panel dataset, which tracks consumer purchases from a wide ranging sample of US consumers at a barcode or universal product code (UPC) level from 2004-2017. As part of the program, consumers were asked to either scan or record the quantities and prices of all goods with a barcode in Nielsen tracked categories. Panelists drop in and out of the program, but the number of consumers is set to be around 40,000 from 2004-2006 and 60,000 from 2007 onward[2].

Nielsen divides products in the dataset into a product hierarchy. At the most aggregate level, products are classified into 10 "departments" (e.g. dry grocery, dairy, general merchandise), which are then broken into around 100 "groups" (snacks, cheese, toys and sporting goods), and further into around 1000 "modules" (potato chips, cheddar cheese, bicycles). At the finest level, we see individual UPCs, which are generated for any new product with meaningful differences from existing products. A new UPC can either represent a substantial change like an entirely new product category or something as minor as a change in the size or color of a product. Purely cosmetic changes in packaging that do not change any characteristics of the underlying product do not generate a new UPC. Having data at this fine level is helpful for analyzing changes in prices, because changes in quality of any product will always generate a new UPC. Therefore, we do not need to worry about changes in quality of existing products causing biased estimates of the cost of living.

In analyzing this data, two major stylized facts quickly jump out. First, consumers change the set of products that they buy quite frequently and second, there is substantial heterogeneity in consumer purchasing behavior. The remainder of this section explores these two facts in more detail.

2.2.2 Frequency of Product Substitution

If consumers purchased largely the same basket of goods over time, calculating a price index would be trivial. We could simply calculate the changes in the cost of an individual consumer's basket over time (as is done in a standard Laspeyres price index). However, if consumers frequently change the products that they purchase, substituting towards cheaper products or newer and better products, we need to make a judgement on the relative values of

[2]because of this discontinuity in sample size, I restrict most of my results to 2007-2017

each product. The data shows that product substitution and product turnover are important features of reality, which suggests that estimates of cost of living that do not account for these changes are not necessarily good descriptions of the costs consumers face.

In particular, if we examine the frequency at which products are added and dropped, we quickly see that assuming a constant product set will give a distorted view of real consumer behavior. Figure 2.1 below gives an idea of the number of products available in each year and the quantities of entering and exiting products. On net, we see a slight increase in the total number of products purchased (about a 7% increase). We also see a high degree of product turnover, with approximately 200,000 of the 750,000 products entering within the last year or exiting the following year.

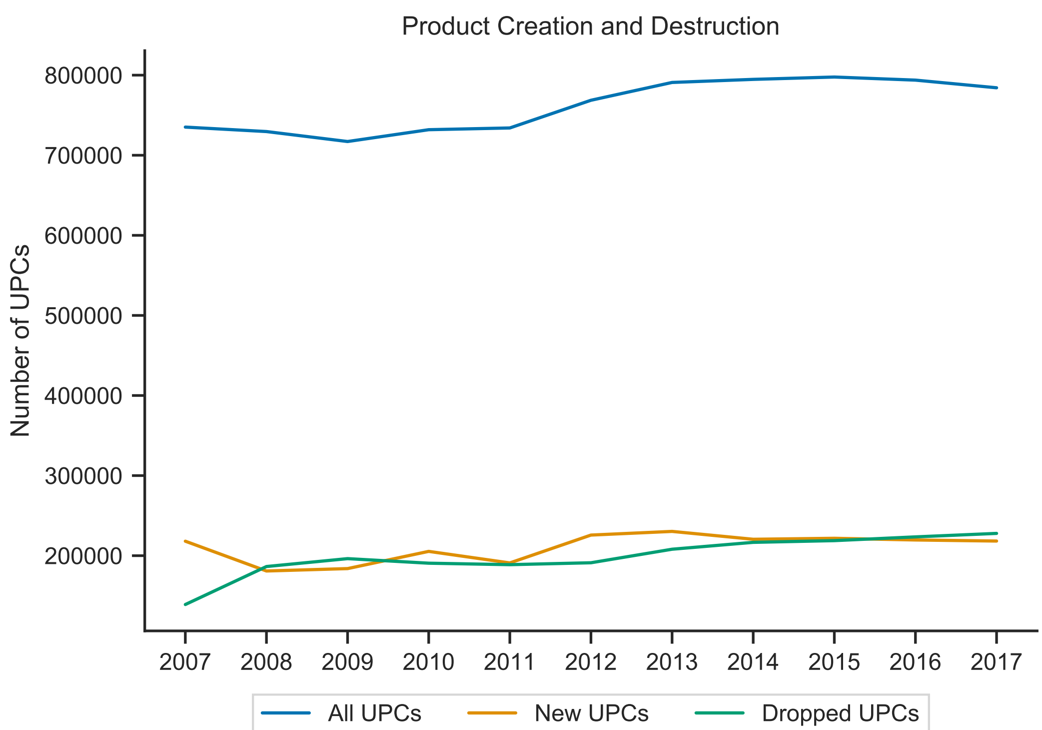

Figure 2.1: Product creation and destruction over ten years. The blue line shows the total number of UPCs purchased by consuemrs in the sample. The orange line shows the number of UPCs that were not purchased in the previous year and the green line shows the number products that were purchased in the previous year but are not in the current year.

It is possible that the products being created, despite being many in number, do not make up a large portion of total expenditure. However, a similar story emerges when we examine the spending patterns of consumers for new and existing products. Figure 2.2 shows the share of expenditure on products that were available in 2007 in each of the following years. We see that the share drops off rapidly, decreasing to around 30% by 2017. In other words, 70% of expenditure on products consumed in 2017 were not purchased at all in 2007. In order to ensure that the effect is not driven by changes in the set of consumers, we restrict the sample for the figure to consumers who remained in the panel over the entire time period. Therefore, for these consumers we can say that their consumption bundle at the beginning of the period was substantially different than their bundle at the end.

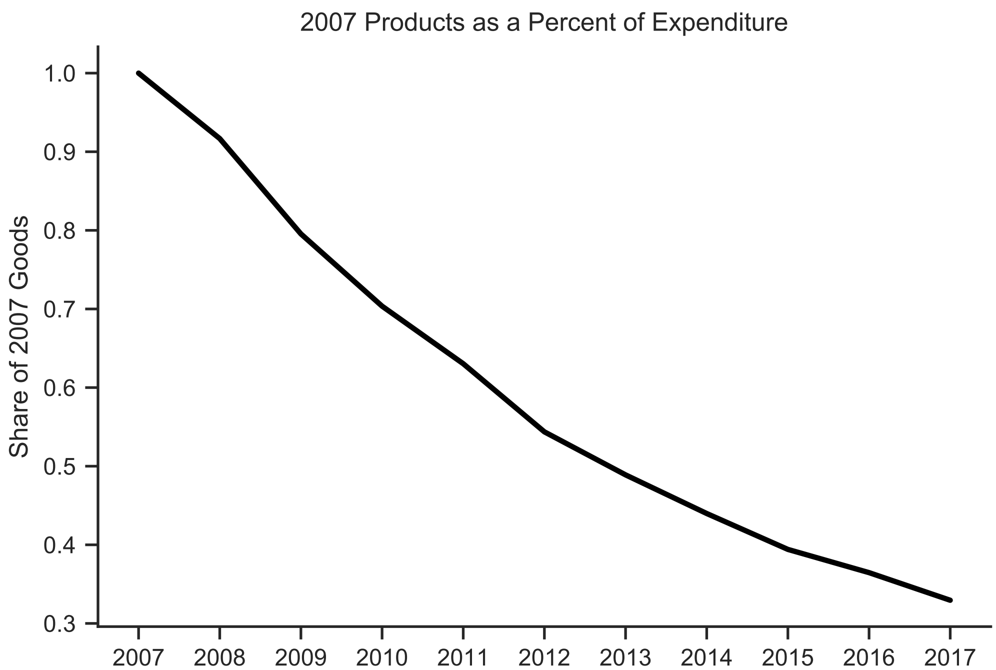

Figure 2.2: The share of expenditure on products available in 2007. The set of consumers is restricted to those who remained in the panel from 2007-2017 (about 10,000 consumers). I fix the bundle of UPCs that these consumers purchased in 2007. Then I calculate the share of total expenditure in each year that this bundle comprises.

There are many potential explanations for the changes in product sets illustrated above. We will not explore these kinds of questions here. Instead, the key takeaway from the graphs should be that the assumption of a fixed consumption bundle could cause problems for the estimation of a price index. If we measure changes in prices of goods that were available in 2007, we won't do a good job explaining the cost of living for a consumer living in 2017.

2.2.3 Heterogeneity in Purchases

Another challenge in estimating an aggregate price index is that not all consumers buy the same set of products. If a consumer hates broccoli, a fall in the price of broccoli isn't going to improve their quality of life at all. For a vegetable lover, the same price drop will have a larger impact. This concern becomes even more pressing when we consider the facts about product turnover above. If new products are only reaching a subset of total consumers, treating them as if they were being consumed by everyone could give misleading results about aggregate welfare.

Looking at the data, it is clear that consumers exhibit large differences in purchasing behavior. Even if we constrict the set of products to the top sellers, not all consumers are making similar decisions about which products they purchase. The histogram below shows the distribution of the percentage of consumers each product reaches. Although there is a fat tail, with a small minority of products reaching the majority of consumers, almost all products sell to less than half of consumers with the typical product selling to somewhere around 10% of all consumers. I restrict the product set here to only include products with at least 50,000 units sold over the sample (about 2000 UPCs). Products that don't sell as much reach an even smaller percentage of consumers.

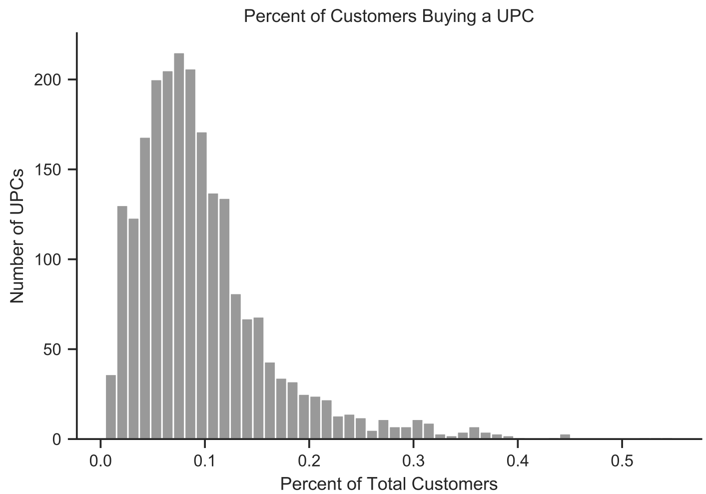

Figure 2.3: Distribution of Goods by Percentage of Customers Reached. For each UPC, the percentage of customers is calculated by dividing the number of consumers who purchased the product by the total number of consumers. I restrict the sample of UPCs to those that sold at least 50,000 units over the entire sample.

With these facts in mind, it is clear we need a model that can both account for a changing set of products while also allowing for wide heterogeneity in consumer tastes. Previous work has dealt with the first question pretty well. However, the next section shows that using these kinds of methods when there is heterogeneity in consumer tastes can give misleading results about the elasticity of substitution between products. Section 4 will then introduce a potential solution to these issues by showing how a density peaks clustering algorithm can be used to pull out consumer heterogeneity before continuing with the elasticity estimation.

2.3 CES Aggregation

To motivate the use of the clustering algorithm to improve CES aggregation, this section reviews results on the aggregation of a logit demand system to a CES representative agent and shows why methods that use the aggregate demand system to estimate elasticities and aggregate price indexes cannot satisfy their identifying assumptions when consumers have heterogeneous preferences.

The benchmark comparison will be to a CES representative agent frequently used in the macro and trade literature (Dixit and Stiglitz, 1977). While other preference specifications can offer more flexibility for adding realistic features, CES has remained a valuable tool for macroeconomists due to its nice aggregate properties. In this framework, the unit expenditure function for the representative consumer is given by

$$P_t = \left(\sum_{k \in \Omega_t} \left(\frac{p_{kt}}{\phi_{kt}} \right)^{1-\sigma} \right)^{\frac{1}{1-\sigma}},$$

where p_{kt} is the price of variety k at time t, ϕ_{kt} is a parameter that could represent either the quality of variety k or the consumer's taste for variety k, Ω_t is the set of all varieties available at time t and σ is the elasticity of substitution across products. This price index gives a measure of consumer welfare as it represents the cost of buying each unit of utility.

The other key result of CES preferences is that the share of consumption on each variety is given by

$$s_{kt} = \frac{(p_{kt}/\phi_{kt})^{1-\sigma}}{\sum_{l \in \Omega} (p_j/\phi_j)^{1-\sigma}}. \tag{2.1}$$

Although CES is analytically tractable, the question remains whether it is a realistic representation of consumer behavior. One nice justification for using CES is that it can represent heterogeneity through a discrete choice problem at the micro level. The next section lays out some of the limitations of that justification.

2.3.1 Logit Aggregation with Taste Heterogeneity

The basic setup of the discrete choice framework in this section is drawn from the standard setup in the literature (Anderson et al., 1992). Assume that a consumer faces a choice between a set of n differentiated products indexed by k. They can choose a single product to consume. The direct utility that consumer i receives from consuming c units of product k is given by

$$\begin{aligned} U_{it} &= u_{ikt} + \varepsilon_{ikt} \\ u_{ik} &= \ln(\phi_k^i) + \ln(c_{ikt}), \end{aligned} \tag{2.2}$$

where ϕ_k^i is the consumer $i's$ taste for good k, c_{ikt} is consumer $i's$ consumption of good k, and ε_{ik} is an i.i.d shock to consumer's preference that captures idiosyncratic variation in consumer tastes. We will assume that all consumers are identical in income and face the same price so that consumption for a consumer that chooses to consume good k receives

$$c_{ikt} = \frac{y}{p_k t}$$

If we assume that the ε shock is distributed according to the Gumbel double exponential distribution

$$Pr(\varepsilon_i < x) = e^{-e^{-(x/\mu+\gamma)}},$$

where γ is Euler's constant ($\gamma \approx 0.5772$) and μ is a positive constant, then it can be shown that the probability of a consumer choosing good k is

$$Pr(k) = \frac{\exp(u_k/\mu)}{\sum_{j=1}^{n} \exp(u_j/\mu)}, \tag{2.3}$$

If we assume all consumers shared the same taste ($\phi_k^i = \phi_k$ for all i), then plugging in the utility function and aggregating across all consumers, the share of expenditure on a given variety k is given by

$$s_{kt} = \frac{(p_{kt}/\phi_{kt})^{-1/\mu}}{\sum_{j\in\Omega_t}(p_{jt}/\phi_{jt})^{-1/\mu}}. \tag{2.4}$$

Therefore, the share of consumption by all consumers in the logit framework corresponds to the representative consumer in Equation 2.1 with the shape parameter of the double exponential governing the degree of substitutability across products. However, the aggregation of logit preferences to CES is a somewhat fragile result. To get the aggregation above, we needed to assume that consumers face the same set of prices and share the same tastes for all goods (apart from the idiosyncratic ε shock). Recent work has demonstrated that the CES representative consumer is not an accurate representation of the underlying logit demand system when there are asymmetric price changes (Tito, 2016). It can also fail when there is taste heterogeneity across consumers.

To see this breakdown, note that when ϕ_k^i differs across consumers, we can still use Equation 2.3 as the probability of each consumer buying each good. However, the share of expenditure on each variety no longer affords easy aggregation since each consumer will potentially have a different probability due to their differing tastes. To go any further, we need to make some assumptions about the distributions of tastes across consumers.

Certainly allowing for heterogeneous tastes in a more flexible way is not a new problem. The mixed logit, or random coefficients, model used perhaps most prominently in Berry et al. (1995) was designed specifically for this purpose. This paper takes a somewhat different approach more akin to the latent class model described in Greene and Hensher (2003) (although the identification of these latent classes will be different here). In a latent class model, consumer's choice probability depends on some unobserved variable or set of variables that classify them into discrete classes. Although the latent class formulation is in many ways less flexible than mixed logit models that allow for continuous distributions of parameters across consumers, it sidesteps the issue of having to define specific distributions of parameters.

To give an example relevant to the data that will be explored later in the paper, we can think of consumers deciding which snacks they want to purchase. Certainly price will play a role as consumers switch to cheaper products that have similar characteristics. However, as we saw in the evidence in section 2.2, consumers do not always substitute across the entire product set. This heterogeneity can be caused by a variety of factors. Some goods might not be available in all areas, different income groups might have different preferences, or individuals could simply have different tastes. A consumer whose preferred snack is popcorn might freely switch between different brands of popcorn, but wouldn't be as quick to change to potato chips. These different tastes could be accounted for in a variety of ways, including narrowing the product set to only focus on obvious close substitutes. Here we will take the approach of making the taste parameter ϕ group specific. In other words, all popcorn lovers will be placed together in a group that have a relatively high average taste for popcorn products.

For now, we shall take as given that we can correctly classify consumers into groups that all share the same taste parameters. If we can successfully perform this classification, and assuming we have sufficient consumers in each group to aggregate, we can clearly see that the share equation specified in Equation 2.1 will hold separately for each group

$$s_{kt}^{i} = \frac{(p_{kt}/\phi_{kt}^{i})^{1-\sigma}}{\sum_{j\in\Omega_t}(p_{jt}/\phi_{jt}^{i})^{1-\sigma}}. \tag{2.5}$$

Taking the ratio of two products and taking logs, we can solve for σ as

$$1-\sigma^{i} = \frac{\ln(s_k^i/s_j^i)}{\ln(p_k/p_j)-\ln(\phi_k^i/\phi_j^i)}.$$

Methods for dealing with the estimation of σ when there are demand shocks will be discussed below, but for now let's assume demand is constant over time for all goods. This assumption means that the elasticity within each group is exactly identified with only share and price data by differencing over time

$$1-\sigma^{i} = \frac{\Delta\ln(s_k^i/s_j^i)}{\Delta\ln(p_k/p_j)}.$$

What if we did not know the groups and instead used the aggregate shares? In principle, we can always find an aggregate taste parameter and aggregate elasticity that would match any data driven by an underlying set of heterogeneous groups, but will these parameters accurately represent the original groups? In many cases, the answer is no. To illustrate the main idea, let's consider an example with two groups of consumers ($i \in \{A, B\}$) and two goods ($k \in \{1, 2\}$). We want to compare the elasticities implied by the shares purchased by each group to the representative agent. To examine the most extreme case, let's assume that consumers in Group A have a much stronger taste for good 1 and consumers in Group B have a much stronger taste for good 2 and the two groups have equal expenditure weights and elasticities. If the difference in tastes is large enough (i.e. as $\phi_1^A/\phi_2^A \to \infty$ and $\phi_2^B/\phi_1^B \to \infty$), we can say

$$s_1^{agg} = \frac{s_1^A + s_1^B}{2} \approx \frac{s_1^A}{2}$$
$$s_2^{agg} = \frac{s_2^A + s_2^B}{2} \approx \frac{s_2^B}{2}$$

since each consumer will consume a much larger share of the good they like more. Assume the price of good 1 changes relative to good 2. Then we can calculate the log change in the relative share for the aggregate consumer as

$$\Delta\ln(s_1^{agg}/s_2^{agg}) \approx \Delta\ln(s_1^A) - \Delta\ln(s_2^B).$$

When the tastes are strong enough, the aggregate consumer is indistinguishable from the consumer who has strong tastes for that good. In this case, they look like consumers from group A for good 1 and consumers in group B for good 2. Without loss of generality, assume $\phi_2^A = \phi_1^B = 1$ and $p_2 = 1$. Then substituting the shares into the expression above and simplifiying produces

$$\Delta \ln(s_1^A) - \Delta \ln(s_2^B) = \Delta \ln(p_1^{1-\sigma}) - \Delta \ln\left((p_1/\phi_1^A)^{1-\sigma} + 1\right) + \Delta \ln\left(p_1^{1-\sigma} + (1/\phi_2^B)^{1-\sigma}\right).$$

Taking the limit of the above expression as $\phi_1^A \to \infty$ and $\phi_2^B \to \infty$, we can see that the difference goes to 0. The next step is to relate this result to the elasticities. We have argued above that the percentage change in the aggregate ratio of the shares goes to 0 as the difference in shares becomes large. Then if we assume aggregate tastes are unchanged,

$$1 - \sigma^{agg} = \frac{\Delta \ln(s_1^{agg}/s_2^{agg})}{\Delta \ln(p_1/p_2)} = 0 \implies \sigma^{agg} = 1.$$

In other words, as tastes for each group of consumers become large, the aggregate consumer looks like they have Cobb-Douglas preferences where expenditure shares are fixed[3]. Figure 2.4 below shows the calculated sigma for different levels of the strength of preference consumers have for each good. When consumers have relatively diverse preferences (when ϕ_1^A is large relative to ϕ_1^B), the aggregate elasticity approaches $\sigma = 1$. Intuitively, if consumers have very strong preference for 1 good over the other, there will only be tiny changes in their overall shares when prices change since taste adjusted prices barely change. The aggregate consumer will remain close to having equal shares across both goods, which makes it look like there is little substitution. On the other extreme, when consumers have equal tastes across both goods, they will both respond in the same way to price changes and the aggregate consumer will be indistinguishable from the individual groups.

[3]In this example they were fixed at 1/2. Had we chosen different expenditure weights for the consumers, the shares of the aggregate consumer for each good would correspond to these weights

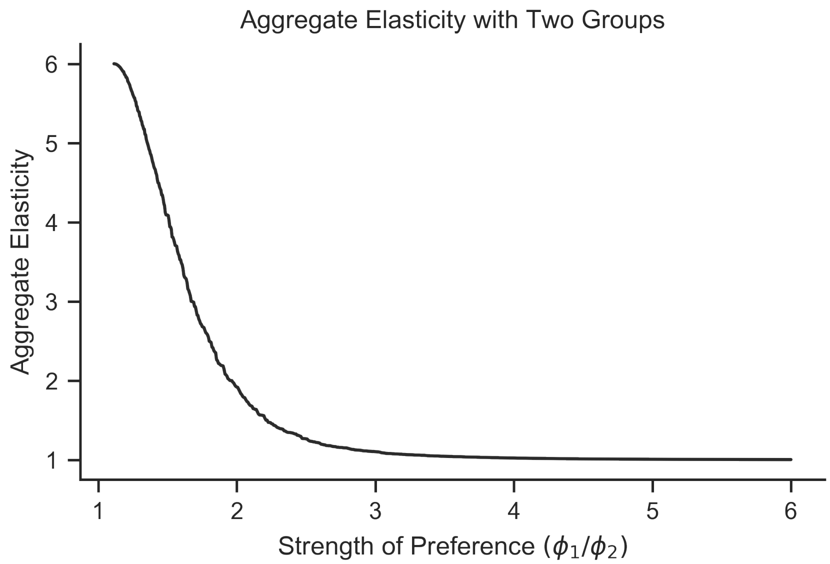

Figure 2.4: The implied aggregate elasticity for different levels of taste differences between consumers. Results shown are for 2 groups of consumers with constant tastes over two goods and the same elasticity $\sigma = 6$. Consumers in group A prefer good 1 at the ratio on the x-axis while consumers in group B prefer good 2 at the same ratio. As the gap in consumer tastes increases, the implied elasticity falls from 6 (the common group elasticity) to 1

In the example above, we assumed aggregate tastes were held constant, meaning any difference between the aggregate consumer and each group had to be absorbed in the elasticity parameter. We could also allow for changes in the aggregate taste parameters, but this flexibility only helps if individual group parameters are already known. In particular, we can always find aggregate taste parameters that satisfy

$$1 - \sigma = \frac{\Delta \ln(s_1^{agg}/s_2^{agg})}{\Delta \ln(p_1/p_2) - \Delta \text{ln}(\phi_1^{agg}/\phi_2^{agg})}$$

for any sigma. Remember that in the extreme case where each group very strongly prefers one good over the other, the percent changes in aggregate shares will be close to zero (shares will stay about constant for any prices). Then if we want to rationalize an aggregate elasticity greater than 1, we would need taste shocks to almost exactly offset price shocks. But without knowing the elasticity for each group, these aggregate shocks cannot be identified from the data and are not helpful in estimating elasticity on aggregate data.

In many ways, the two good, two group example is a special case. However, the intuition carries over (with some caveats) to larger sets of groups and varieties. The appendix gives a longer discussion and shows that the aggregate elasticity looks low when the mean price change of goods for which they have a strong taste is different from the mean price change of goods for which they have less taste. While this is certainly not guaranteed, if we think that products that a consumer prefers share some common characteristics, it is reasonable to believe that they would also tend to have more correlated prices. In other words, a price increase of a popcorn product is more likely to correspond to increases in other popcorn products than an increase in potato chip products. As a result, we expect to see a lower elasticity for an aggregate representative agent than for the groups it represents.

It is not necessarily a problem that aggregate elasticities are below elasticities for individual groups. If all we care about is reproducing the aggregate share data that we see, there is no problem using a representative agent to describe the sum of the individual groups. However, if we want our representative agent to accurately represent the *welfare* of each group, we run into trouble. In fact, it is possible that the welfare implications gleaned from the representative agent lie outside any weighted average of the individual groups. In the next section, I will look at how the representative agent can give a misleading picture of how product innovation affects welfare.

2.3.2 Aggregate Price Indexes with Heterogeneity

A nice property of CES preferences is that they give an easy way to deal with a changing product set. Feenstra (1994) showed that the change in the aggregate price index in a standard CES framework can be decomposed as

$$\frac{P_t}{P_{t-1}} = \frac{P_t^*}{P_{t-1}^*}\left(\frac{\lambda_t}{\lambda_{t-1}}\right)^{\frac{1}{\sigma-1}}, \tag{2.6}$$

where P^* is the standard CES price index over goods common to periods t and $t-1$ and λ is the fraction of expenditure on these common goods. More formally, we can define λ_t and

λ_{t-1} as

$$\lambda_r = \frac{\sum_{k\in\Omega_{t,t-1}} p_{k,r}x_{k,r}}{\sum_{k\in\Omega_r} p_{k,r}x_{k,r}} \quad \text{for } r \in t, t-1,$$

where Ω_r represents the set of goods available at time r (so $\Omega_{t,t-1}$ are goods available in both t and $t-1$). If the share of common goods expenditure is low in period t (λ_t is low), then people prefer to buy newly introduced products rather than existing products and the price index falls. If the share of common goods expenditure is low in period $t-1$, then products that people liked in period $t-1$ have disappeared in period t, which increases the price index. In other words, product entry will tend to decrease the price index while product exit will increase it.

The elasticity parameter σ plays a crucial role in determining the effect of product creation and destruction. If the new products coming in are very substitutable (σ very high) for existing products, then they do not add much to consumer welfare since they could just as easily be substituted for products that were already available. However, if products are highly differentiated, then adding new choices for consumers will have large benefits. This result is closely related to the standard "love of variety" result that comes from CES preferences. Other things equal, a consumer prefers to consume a larger set of products, and this preference is stronger the more differentiated those products are.

This decomposition is only exact when tastes for common goods are unchanged over time. Because this assumption is quite strong, recent research has worked to lessen the requirements by altering the price index. Redding and Weinstein (2019) show that we can allow for taste shocks using an extension of the Feenstra price index in Equation 2.6 that they call the "CES Unified Price Index" (CUPI). The index can be written

$$\frac{P_t}{P_{t-1}} = \left(\frac{\lambda_t}{\lambda_{t-1}}\right)^{\frac{1}{\sigma-1}} \left(\frac{\tilde{p}_t^*}{\tilde{p}_{t-1}^*}\right) \left(\frac{\tilde{s}_t^*}{\tilde{s}_{t-1}^*}\right)^{\frac{1}{\sigma-1}}, \tag{2.7}$$

where a tilde indicates geometric mean and an asterisk denotes the mean is over varieties common to both periods. The first two terms of this expression are essentially the same as those in the Feenstra index, capturing the change in the prices of common goods and the introduction of new products, but the third term, which measures the change in the dispersion of shares, is new and allows the price index to account for taste shocks as long as they have a constant geometric mean over time. The assumption of a constant geometric mean rules out any pure increase in overall tastes (for example if goods are increasing in quality on average over time) so it is still somewhat restrictive. However, it does allow for certain goods to become more popular at certain times (due to seasonal effects, trends, etc.), which is a clear improvement over assuming tastes need to be held constant in the common good set.

A full analysis of the assumptions and properties of the price index is in Redding and Weinstein (2019) so I will not repeat it here, but it is worth reiterating the intuition for why the new term captures changes in tastes. Imagine prices of all goods and all tastes were the same. Then CES preferences imply equal shares across all goods. However, when products are substitutable ($\sigma > 1$), consumers would be better off if there were dispersion in (taste adjusted) prices because they could substitute towards cheaper products (or products they like relatively more). Holding the mean taste parameter fixed, the consumer is better off as the dispersion of tastes increases, which will increase the dispersion of shares, lowering $\tilde{s}$ and therefore lowering the price index. As with new products, the same size increase in dispersion has a larger effect when σ is smaller.

Returning to our setup with distinct groups of consumers, we can compare the price indexes that result from each group and from the aggregate representative agent. As discussed above, if different groups of consumers have different tastes, the aggregate shares will in general have smaller percent changes compared to within group shares, leading to a smaller estimate for the elasticity when using aggregate data. In the price index, two terms depend on the elasticity. Either can potentially cause the change in the aggregate price level for an aggregate representative agent to be different from the individual groups. However, the dispersion in shares will naturally be lower for preferences with lower elasticity (which is why we get a lower elasticity in the first place). There is not such a tight relationship with the share of new goods in expenditure. It is entirely possible to have high or low values for λ_t/λ_{t-1} for a given σ

Figure 2.5 shows the effects of having a low estimate for the aggregate elasticity on aggregate price indexes. In this example, I let prices for common goods fluctuate around a mean of zero and introduce new products to set $\lambda_t/\lambda_{t-1} = 0.99$. This value of net product creation implies a small amount of product creation (less than we will see in the real data), but it is enough to open a gap between the grouped data and the aggregate representative agent. Consistent with the discussion above, the lower elasticity estimate for the aggregate agent means that new products due more to increase welfare (and therefore reduce the price index at a faster rate).

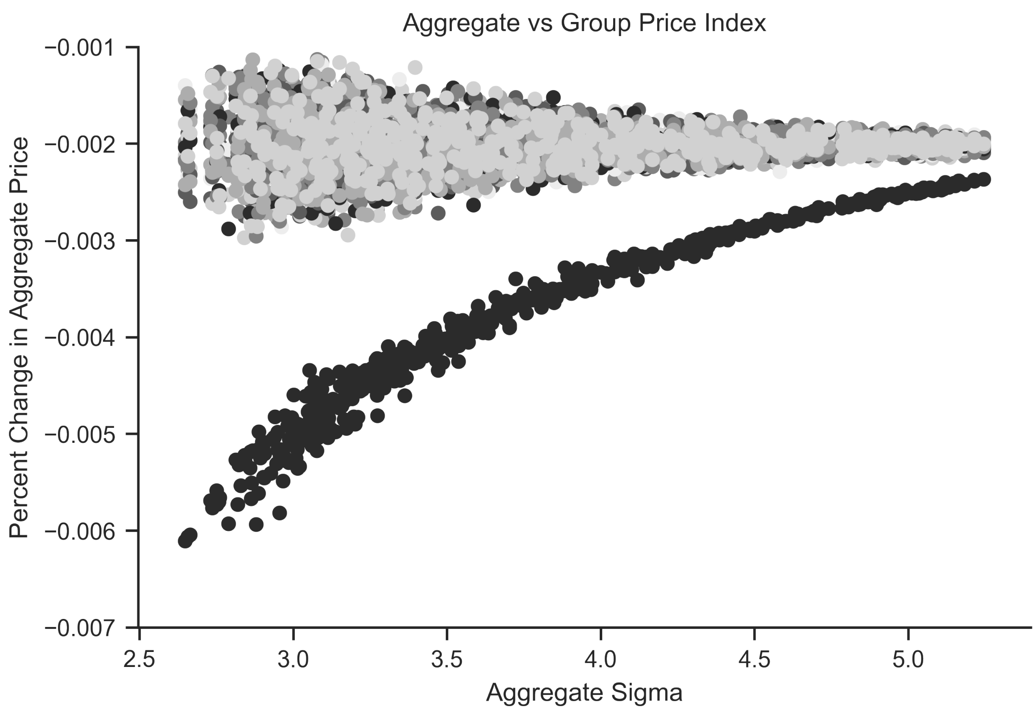

Figure 2.5: Estimated price index on the aggregate data (black) against individual groups (grays). Results are generated from 10 groups of consumers buying 200 products. Each group has a strong taste ($\phi = 10$) for 10 of the products and $\phi = 1$ for all others. Each group shares a common elasticity $\sigma = 6$. Different aggregate elasticities are generated by varying the correlation of price changes within taste sets. Results are plotted after 500 replications.

This example illustrates the main problem that this paper aims to solve. If we use aggregate data to estimate changes in the cost of living, we might give excessive weight to product creation and destruction in the aggregate price index. Although the aggregate estimates are supposed to "represent" some sort of average of the individual consumers, the aggregate estimates actually lie below *all* the groups. Using an aggregate representative agent would produce an estimate of inflation that is lower than the one actually experienced by any group of consumers.

2.4 CES Price Index Using Density Peaks Clustering

To try to solve the problem of taste heterogeneity, I apply the Density Peaks Clustering algorithm introduced in chapter 1. The goal is to improve estimates of an aggregate price index compared to previous work, which has taken a representative agent approach. I perform all analysis at the product group level, which includes relatively broad categories such as "snacks", "candy", and "milk" (a list of all product groups is provided in the appendix). For the final price index estimates, I aggregate groups by expenditure to form an average price index for the whole economy, but all clustering and elasticity estimation is done within product groups.

The Density Peaks Clustering algorithm works by looking for groups of consumers who buy a similar set of products. As discussed above, observable characteristics are often insufficient to explain differences in consumer tastes. However, if we group consumers who have different tastes, we saw that we could get a bias in estimates of elasticity and the impact of product variety. The method introduced in chapter 1 offers a way around this problem. Through simulations on a logit demand system, chapter 1 showed that the DPC clustering algorithm could effectively discover groups of consumers if they had different tastes on average (as in section 2.3 of this chapter).

I will present the results in three parts. First, I discuss the clustering itself. I demonstrate that the clustering successfully separates consumers who have relatively larger purchase shares for some goods compared to consumers in other clusters. I also present some results on whether the clusters can be predicted using only observable characteristics. In general, there is some correlation of observables across clusters, but not enough to fully explain the resulting clusters. In other words, the clusters represent heterogeneity in tastes that goes beyond observable features of each consumer. I then move to the results on the estimation of elasticities. I describe the estimation procedure and show that the predictions of the theory presented in section 2.3 are mostly confirmed. For product groups where more than one cluster is found, a strong majority of elasticities within a cluster are higher than those estimated on aggregate data. Finally, I calculate the implied price index from the estimated elasticities using the methodology outlined in section 2.3. Clustered price indexes exhibit substantial heterogeneity across clusters, but are almost uniformly higher than the corresponding price index for a representative agent.

2.4.1 Clustering Results

Before getting to the results of running the clustering algorithm on the consumer data, we need to discuss a few choices in bringing the method to the data. In particular, one important decision I made when applying the DPC algorithm is to assume that consumer taste varies over *brands* rather than individual UPCs. One reason for this choice is practical. Many individual UPCs have very few purchases and some only exist in the dataset for a short period of time before exiting. Including these products adds additional shares that are close to zero for each consumer, which pushes the distances of all consumers closer together, making it harder to pull out heterogeneity. However, the assumption also has intuitive appeal. If a consumer usually buys a six pack of Coke over a six pack of Pepsi, they are probably going to prefer a 12 pack of Coke over a 12 pack of Pepsi as well. The difference in brand is often more interesting than the difference in UPC, since the latter can often be relatively minor changes[4].

[4]One decision becomes whether to include generic grocery store brands. In the data, all store brands are grouped into one to hide the identities of individual chains. There are arguments for removing these categories from the clustering since the actual products will differ across stores. I take the position that store brands are more similar to each other than they are to brand name products. Therefore, a person buying a significant portion of store brands is a difference in taste that we want to account for and I choose not to remove this information.

However, even aggregating products to the brand level, there are still many brands that are purchased by few consumers or are a negligible share of most consumer's expenditure. Once again, including these products will make consumers look more similar since they are close to zero for everyone. To eliminate these smaller brands, I put a minimum on the sum of expenditure shares across consumers (set to 50 in the benchmark result). Under this condition, included brands are either a large share of expenditure or are a relatively small share but purchased by many consumers (so the sum is still high). With this condition, I can keep the list of included brands more manageable.

Because the ability to differentiate consumer tastes rests on the assumption that consumers make a number of consumption decisions over time, I also restrict the households included in the sample to those that have made at least 20 purchases over the entire sample. After restricting the household and brand set, the remaining data makes up around 3/4 of expenditure for most groups. However, a few smaller product groups end up with significantly less expenditure. I exclude these groups from the clustering.

The next choice to make is what time period to cluster over. Because I need a relatively long history of purchases for the clustering to be effective (as shown in chapter 1), I use the entire sample. This time period only makes sense if consumer's average tastes do not change much over time. While this assumption is probably not exactly true, there are certainly many tastes that won't change over an approximately 15 year period, especially when we consider brands rather than individual products. I have attempted specifications with clustering within a year or within a quarter, but the number of datapoints drops to a point that raises concerns, especially considering the inherent noise in the data.

Finally, the choice to estimate elasticities at the group level reflects a balance of a couple factors. If clustering were done at a more aggregate level, there would likely be much more heterogeneity in tastes, as tastes may vary both within groups and across groups. Had I instead used a finer level of aggregation (Nielsen provides one finer level of aggregation called "modules"), then there are many fewer observations for each module, which makes estimation of elasticities less precise. Moreover, other papers that have used this dataset have also settled on the group level as the most logical choice (Redding and Weinstein, 2019). By keeping this choice consistent, it is easier to compare results.

Due to the high number of product groups, it is impossible to show results for them all here (the appendix gives some summary statistics on the number and size of clusters). Instead, I will focus on a single product group to demonstrate what typical results look like. I will highlight the product group "Snacks" because of its highly familiar brands (Lay's, Dorito's, Orville Redenbacher).[5]

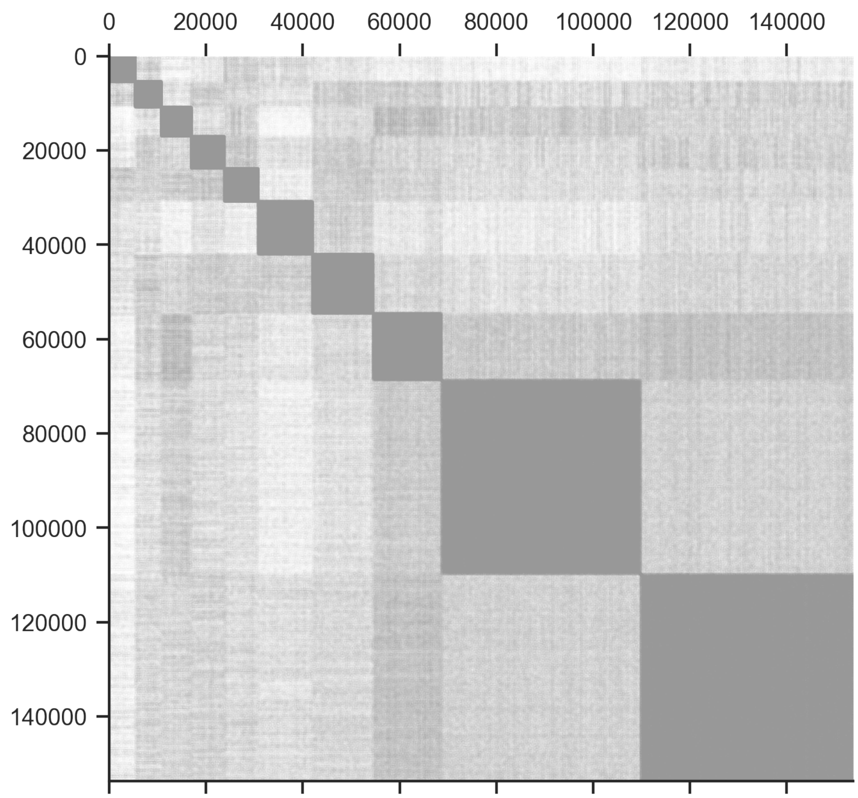

Figure 2.6: KNN block partition of the "snacks" product group sorted by cluster size. 10 clusters were uncovered by the DPC clustering algorithm. Each point on the graph is a nearest neighbor pair, with darker areas sharing many nearest neighbors.

Following the methods of chapter 1 for simulated data, Figure 2.6 shows a block partition

[5]Nielsen restricts the use of brand names. Since specific brand names are not especially important to the analysis, they are omitted in the data below.

of the clusters uncovered from the DPC algorithm on the purchase data in the snacks product group. As in chapter 1, the block partition plots each nearest neighbor weighted by their relative similarity. If consumers within a group were much more similar than they are to consumers outside the group, we would expect to see more shading along the diagonal. Unsurprisingly, there is much more overlap across the resulting clusters than in the simulated data since the real world isn't as well separated as an artificial simulation. However, clusters are still significantly more related to other points within their own cluster than they are to points outside their cluster. I restrict the size of the clusters to contain at least 5000 consumers to ensure there is still plenty of data to run the estimation. Therefore, the smallest clusters have about 5000 consumers while the largest in this case have about 30,000.

Again following the analysis from the simulations, we can also look at the difference between the aggregate shares and the shares within each group to make sure we are actually partitioning the space based on differences in consumer purchasing behavior. Although the data consists of share information for about 60 different brands, I plot the top 12 here to make it easier to see the differences.

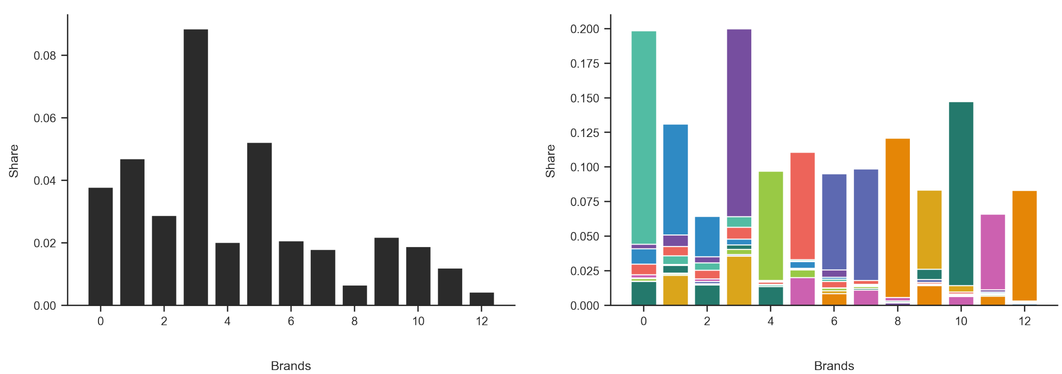

Figure 2.7: Aggregate shares (left) vs clustered shares (right) for the product group "snacks". Each color corresponds to a cluster, and clusters are stacked in order so that all colors are visible.

The results show that the clustering algorithm generally looks for groups that disproportionately prefer one brand over the others. Each cluster has one or two favorite goods that they consume significantly more. While I cannot reveal specific brand information, the results are highly intuitive. For example, notice that brands 8 and 12 are not very popular in aggregate, but are a large portion of expenditure for one of the clusters. It turns out that these brands correspond to major brands of energy bars, which are quite similar to each other but not as similar to the other brands in the product group (chips, pretzels, popcorn, etc.). It makes sense then that consumers who prefer brands 8 and 12 would frequently substitute between these two brands, but not so much with other brands in the group.

A natural question is whether the clusters we have found could be explained by observable characteristics. One way we can look at this question is to start with the clusters and check whether the distribution of observables is different across clusters. If there are substantial differences in the distributions of observables, then we could in theory replicate the clusters found using the DPC algorithm by partitioning on observable characteristics. Previous research suggests that high and low income consumers face different inflation rates in part because they consume different baskets of goods (Jaravel, 2019). However, it does not appear that the clusters estimated from the DPC algorithm are driven by these differences.

The figure below repeats Figure 2.7 using income groups rather than our generated clusters (still for the snacks product group). Although there are some differences in the relative magnitudes, the general shape of distribution is largely the same as the aggregate purchases. In particular, note that the scale of the brand distribution is much closer. For example, one of our clusters had a 20% sale of the first brand, while none of the income groups consume more than 5% of expenditure on that brand. Looking at the correlations in expenditure shares across income groups, all of the 19 income categories consume largely similar brands to the aggregate (> 0.9 correlation). If we do the same exercise with our density peaks clusters, the correlation drops substantially (becoming negative in some cases). In other words, consumers across different income groups are much more similar than consumers across the clusters generated from the DPC algorithm. Although the larger clusters are more correlated to the aggregate, they are still less similar than any of the income groups.

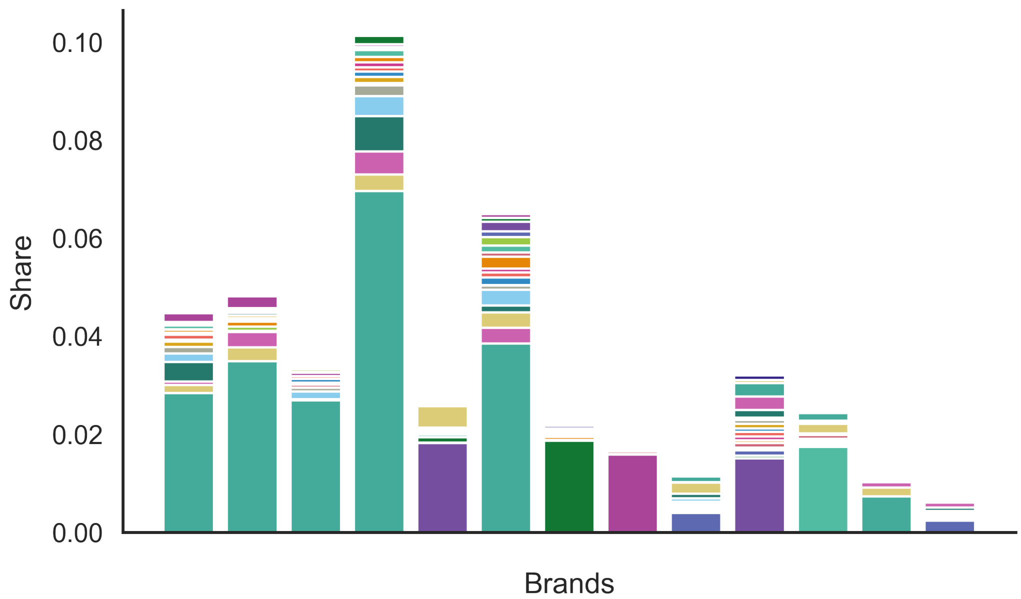

Figure 2.8: Income group shares for the product group “snacks”. Each color corresponds to a different income group, stacked in order so that all colors are visible. Compared to Figure 2.7, we see a much closer relationship between income groups and the aggregate distribution.

To further rule out the idea that the clusters generated by the DPC algorithm could be predicted by observable characteristics, I set up a multinomial logit regression to try to predict the probabilities of each consumer being in a cluster given their observable characteristics. More specifically, for each product group I let the cluster a consumer belongs to, C_i, be a categorical dependent variable and setup the multinomial logit regression as

$$\ln \frac{Pr(C_i = c)}{Pr(C_i = 0)} = \alpha_c + x_i'\beta_c,$$

where β is a vector of coefficients that maps the vector of characteristics (x_i') to the odds ratio of being in cluster c over cluster 0. I try a number of specifications for the observable characteristics and then predict each consumers cluster by choosing the cluster with the highest predicted odds ratio. I then compare the success rate (percentage of consumers placed in the correct cluster) with the rate that would have occurred by random chance. The table below gives some statistics about the success rate across product groups (using income, race, region, age and presence of children, and education as the independent variables)

Table 2.1: Difference in percentage predicted by logit model and random chance

Minimum	Maximum	Median	Mean
-0.05%	21.5%	0.7%	2.0%

In almost all product groups, the prediction model adds essentially no information when compared to a random assignment. The difference in success rates exceeds 10 percentage points in only 3 of the product groups. The three groups where it appears to do well are flour, sugar, and fresh meat. I do not have a strong intuition for why these product groups can be predicted using observable characteristics, but the larger takeaway is that in most product groups, the algorithm is not simply separating consumers based on their demographics.

2.4.2 Elasticity

Now that we have generated clusters for consumers that share similar preferences, the next step is to estimate elasticities within each cluster and compare them to those of the aggregate. Again we will work within product groups, but now we will use the entire set of UPCs rather than aggregate brands, the assumption being that consumers will be likely to substitute to similar products produced under the same brand name. In other words, while it makes sense to aggregate to brands for consumer *tastes*, it makes more sense to use the product level if we want to see whether consumers substitute to cheaper products within a brand when prices change.

To estimate the elasticities, we follow the method introduced by Feenstra (1994). A full discussion of the method is left to the appendix, but I will discuss the intuition here. The method attempts to deal with the endogeneity problem inherent in estimating elasticities without needing to find instruments for price that are uncorrelated with demand shocks. Instead, we assume that *double differencing* the share and price data will help avoid the endogeneity problem altogether. We take a difference with respect to time and with respect to the geometric mean across all products. To identify the elasticities, we need to assume that the demand and supply shocks are orthogonal and heteroskedastic. This process gives us a set of moment conditions (one for each good), which we stack and estimate using GMM (as in Broda and Weinstein (2006)).

Before running the estimation, we need to define the set of common goods. Following Redding and Weinstein (2019), I will use products that exist for at least six years and have existed for at least four quarters and don't disappear for at least four more quarters to avoid the volatility inherent when new products enter.

Estimates for the aggregate elasticity are in line with those estimated in previous work. A full list of estimates is presented in the appendix. More important than the absolute size of these estimates is the comparison to the estimates within clusters. Here the results of the theory are generally confirmed. In almost all of the product groups (excluding some with too few observations to effectively cluster and estimate), the average elasticity within the clusters is higher than the elasticity from aggregate data (for both expenditure weighted and unweighted average).

Figure 2.9 shows the results across all product groups. The first graph shows each of the clustered estimates compared to the aggregate (shown by the black line). As we can see from the figure, most of the estimated cluster elasticities lie above the aggregate estimate. An important note is that I did not constrain the elasticities within clusters to be equal. Therefore, this exercise on real data is not quite a perfect match to the examples given above in the simulations. This difference can make it difficult to disentangle the effects that come from aggregation vs those that are mainly driven by true differences in elasticities across clusters. In theory it would be possible to run the estimation assuming a constant elasticity across clusters, but it seems likely that there actually are differences in elasticity for consumers with different tastes. If consumers are buying products with different characteristics, it's definitely possible that some groups have access to more closely related goods, while some prefer more highly differentiated goods. This effect leads to some cases where the aggregate elasticity is higher than some of the lower clustered groups even if the aggregation bias is downward, but in general the results are consistent with the idea that the representative agent shows less substitution.

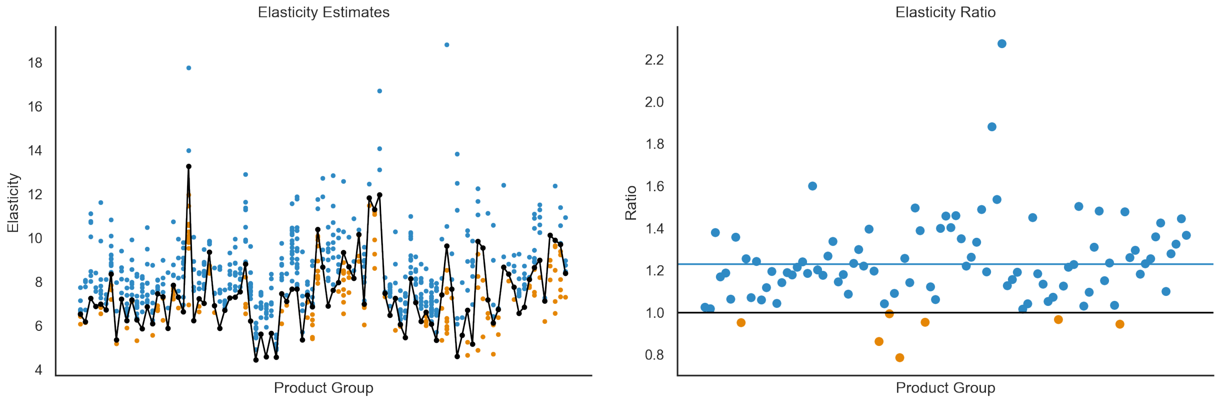

Figure 2.9: The graph on the left shows estimated elasticities for the aggregate (black line) and for each cluster. Clustered points are colored blue if greater than aggregate and orange if less than aggregate. The graph on the right shows the ratio of the expenditure-weighted average of the clustered elasticities to the aggregate elasticity. Points are colored blue if the expenditure-weighted average is above the aggregate and orange if below.

On average, the clustered estimates are about 20% higher than the aggregate (7.1 for the aggregate and 7.8 for the clustered) and we can see from the graph on the right that only 7 of the 95 product groups compared have clusters with average elasticities lower than the aggregate estimate. This result is consistent with the earlier theoretical discussion (and the discussion in the appendix).

2.4.3 Price Index

Finally, given the elasticities in each sector, we can estimate the price indexes within each cluster. As discussed in section 2.3.2, to allow for demand shocks to change the value of the price index we can adopt the formulation of Redding and Weinstein (2019)

$$\frac{P_t}{P_{t-1}} = \left(\frac{\lambda_t}{\lambda_{t-1}}\right)^{\frac{1}{\sigma-1}} \left(\frac{\tilde{p}^*_t}{\tilde{p}^*_{t-1}}\right) \left(\frac{\tilde{s}^*_t}{\tilde{s}^*_{t-1}}\right)^{\frac{1}{\sigma-1}}. \tag{2.8}$$

Prices and shares are observed and we have already estimated the elasticity parameters so all that remains is to plug in the data. I take the time differences as four quarter differences to try to avoid seasonal effects as much as possible. The graph below shows the resulting price index over the period 2007-2017. [6]

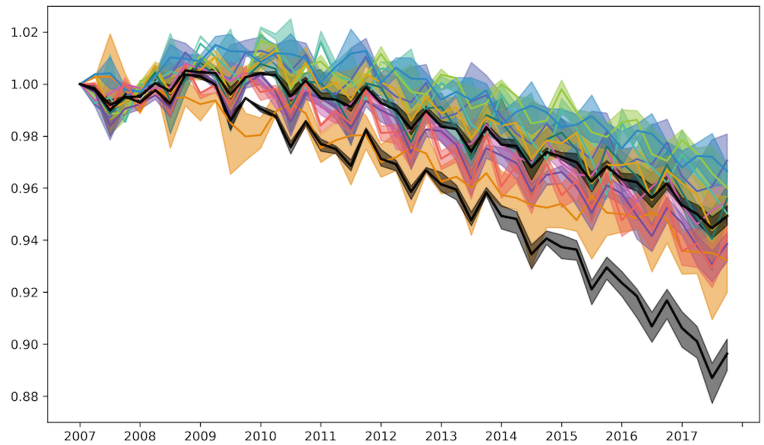

Figure 2.10: Price indexes generated by Equation 2.8 for the product group snacks. The lower black line represents aggregate data (all consumers), while the colored lines represent each cluster's index (with the upper black line showing the average). The range represents a 95% bootstrapped interval from 100 repetitions.

Figure 2.10 shows the results of these price index calculations. The aggregate shares (black line) show a larger decrease in cost of living over the 10 year period than any of the individual groups, as we expected from the theory and the elasticity results. Although some product groups do have individual clusters with lower price index than the aggregate, as we discussed above, these deviations could be driven by product groups whose clusters have different values for the elasticity of substitution (so that the aggregate elasticity lies in between).

Repeating the exercise from the elasticity estimates with inflation numbers, Figure 2.11 shows a comparison between the annual inflation rates implied by the aggregate data vs the clustered estimates. Once again, most of the clustered estimates lie above the aggregate, suggesting that the representative consumer approach understates the true inflation rate compared to the individual clusters. Taking averages of the inflation rates

[6] The graph starts in 2007 because the number of consumers changed from 40,000 consumers in the panel from 2004-2006 to 60,000 from 2007 on). It is possible that this change could result in product entry rates higher than the actual ones due to the changing composition rather than actual introduction of products.

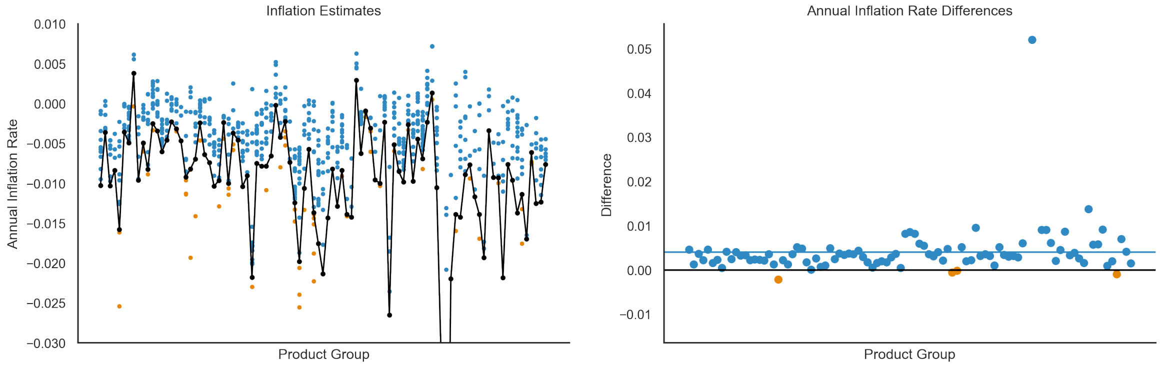

Figure 2.11: The graph on the left shows estimated annual inflation rates for the aggregate (black line) and for each cluster. Clustered points are colored blue if greater than aggregate and orange if less than aggregate. The graph on the right shows the ratio of the expenditure-weighted difference between the clustered inflation rates and the aggregate elasticity. Points are colored blue if the expenditure-weighted average is above the aggregate and orange if below.

Finally, to get a sense of an average measure of cost of living across all product groups Figure 2.12 shows the expenditure weighted average for all product groups at the aggregate level (blue) vs the expenditure weighted average across all the clusters in all product groups. The differences in these price indexes over the ten year period implies a difference in average annual inflation rates of around half a percentage point.

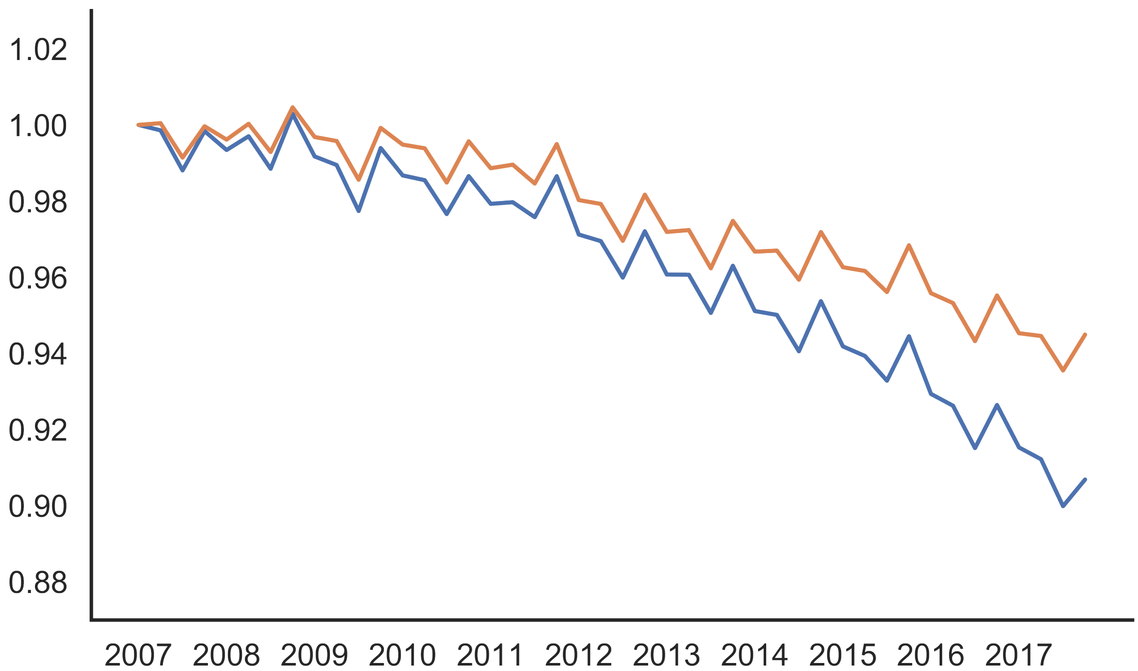

Figure 2.12: Expenditure weighted average across all product groups for the aggregate representative agent (blue) and the expenditure weighted average of all clusters (orange)

Note that this finding does not reverse the results of earlier papers. The clustered index is still far below estimates given by the CPI or PCE index that don't take into account new products and product substitution as well (which average about 1.5% per year). However, it does suggest that those results may have gone a bit too far. Because they assign an elasticity parameter that represents the aggregate data, but not the within cluster data, they attach too large an impact to product innovation in reducing the cost of living. Within a cluster, consumers are more likely to substitute to other products they like and using a representative agent hides this effect.

2.5 Conclusion

Estimating a price index to represent a diverse population of consumers will never be an exact science. There are simply too many sources of heterogeneity among consumption bundles for one statistic to adequately represent each consumer's unique situation. However, we can do our best to account for as many of these sources of heterogeneity as we can. This paper proposed one possible method by using machine learning to cluster consumers into groups that share similar purchase histories. Since these clusters substitute more freely between products, we found that the inflation rate faced by the representative agent was lower than the average across clusters. There is certainly a lot of room to improve the estimation even further. Clustering algorithms could also be performed on the product side of the data to find sets of products that are purchased by the same consumers to help inform product nesting. The methods could also be applied in an international trade context, where the elasticity of substitution plays an important role in determining the gains from trade. Dealing with heterogeneity will never be a trivial problem, but with advances in data availability and analysis, we are getting closer to more accurate answers.

2.6 Appendix I: Aggregation Bias

This appendix provides more intuition on how aggregation can affect estimation of elasticity. While the 2 good, 2 group example in subsection 2.3.1 made it relatively easy to see why aggregation would tend to push down the estimated elasticity, the problem gets substantially more difficult even when moving to 4 goods. Before discussing how the logic changes in more complicated examples, it is worth taking a step back to discuss the problem more theoretically. As noted by Redding and Weinstein (2019), we can always recover σ in CES preferences in a closed form as

$$\sigma = 1 - \frac{\sum_{k\in\Omega_t} \zeta_{kt}\left[\ln\left(\frac{s_{kt}}{\tilde{s}_t}\right) - \ln\left(\frac{s_{kt-1}}{\tilde{s}_{t-1}}\right)\right]}{\sum_{k\in\Omega_t} \zeta_{kt}\left[\ln\left(\frac{p_{kt}/\phi_{k_t}}{\tilde{p}_t}\right) - \ln\left(\frac{p_{kt-1}/\phi_{kt-1}}{\tilde{p}_{t-1}}\right)\right]},$$

where ζ is a set of positive weights adding to 1. In estimating σ, many methods use the Sato-Vartia weights

$$\omega_{kt} = \frac{\frac{s_{kt}-s_{kt-1}}{\ln s_{kt}-\ln s_{kt-1}}}{\sum_{l\in\Omega_t} \frac{s_{lt}-s_{lt-1}}{\ln s_{lt}-\ln s_{lt-1}}}.$$

As Redding and Weinstein (2019) point out, if tastes are unchanged for the set of varieties, then σ is identified for *any* set of positive weights adding to one. On the other hand, if tastes change over time then each set of weights will produce a *different* estimate for σ. This problem becomes amplified when we aggregate consumers with different tastes. As we discussed in subsection 2.3.1, even if all consumers have unchanging tastes individually, when we aggregate, we would need to use a different taste depending on which price moved and by how much. In other words, the assumption of unchanged tastes over time is almost certainly violated.

However, the two good example hides some of the complexity of the problem. When there are only two goods, an increase in the relative price of one good always implies a decrease in the relative price of the other. As we discussed, a high elasticity of substitution would normally imply large responses to price changes. However, when consumers have strong tastes for one good or the other, aggregate shares will change much less than they would if they had the same taste. Since we cannot account for these taste differences, instead the lack of substitution across products is attributed to a lower elasticity.

To motivate a discussion of the complexity in adding additional goods to the question, Figure 2.13 plots the unusual looking distribution of possible elasticity estimates from a simulation with four goods (I will refer to goods using numbers 1-4) where each group has a stronger taste for two of the four goods (group A prefers 1 and 2, B prefers 3 and 4). In this exercise, I run a 2 period simulation with every possible combination of price changes of the four goods (using 20 possible prices for each ranging from an 80% decrease to a 500% increase.

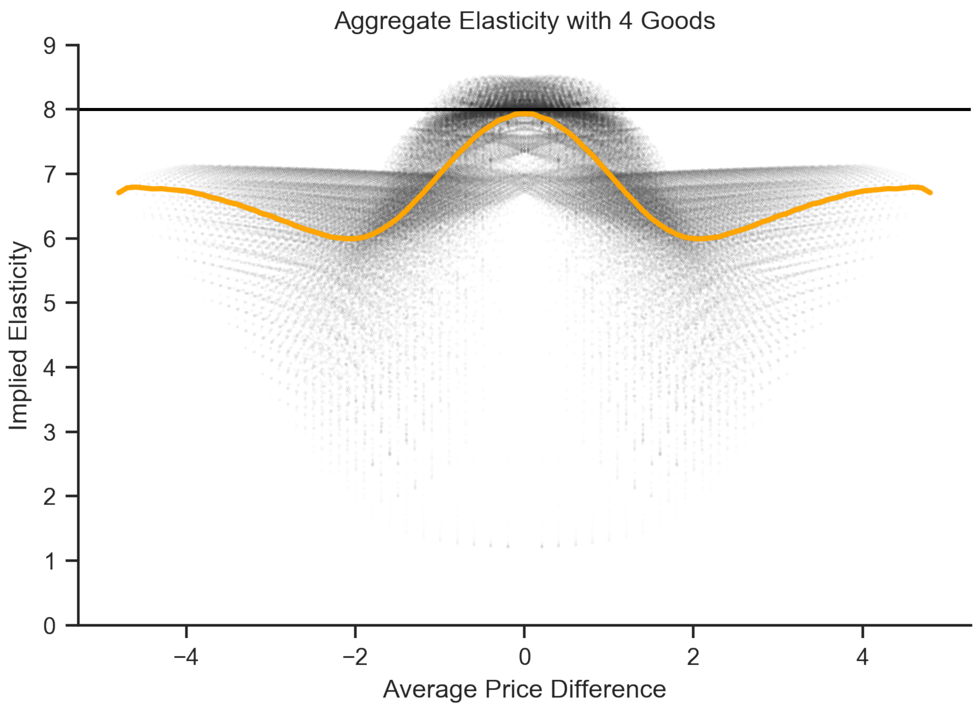

Figure 2.13: Estimated aggregate elasticity against price differences. Price difference is the difference in the geometric mean of goods 1 and 2 vs 3 and 4 $((p_1p_2)^{1/2} - (p_3p_4)^{1/2}$ Results are generated from 2 groups of consumers buying 4 products. Each group has a strong taste ($\phi = 2$) for 2 of the products and $\phi = 1$ for all others. Each group shares a common elasticity $\sigma = 8$ shown by the black line. The orange line plots the mean conditional on the average price difference.

The graph shows the difference between the average (geometric mean) relative price of goods 1 and 2 compared to 3 and 4. For example, a value of 0 means that the average price of good 1 and 2 is equivalent to the average price of 3 and 4. A value of 2 means that the average price of goods 1 and 2 is twice as high as the average price of goods 3 and 4. To understand why this variable is an interesting one to condition on, let's consider a few cases.

First, imagine goods that share a common taste structure always change prices together. In this example, goods 1 and 2 and goods 3 and 4 have equivalent tastes. If they have the same price, consumers with CES preferences will always consume an equal amount of 1 and 2 and an equal amount of 3 and 4. In this case, the problem collapses to the two good case. If we add the consumption of goods 1 and 2 and the consumption of 3 and 4, all the results discussed in the two good case hold up and so we would always expect the aggregate elasticity to be below the elasticity of either group (in this case 8). In this case, the average price of goods 1 and 2 must always be different than goods 3 and 4 (unless they all have the same price).

If we let the relative prices between similar taste goods change, the problem becomes much harder. There are now two effects, which I will call "Like substitution" and "unlike substitution". Like substitution refers to substitution across goods that share the same taste structure. Unlike substitution occurs when consumer substitute goods they like for goods they don't like as much or vice versa. In the example of the preceding paragraph, consumers only exhibited unlike substitution since the relative prices between like goods were unchanged. Unlike substitution for the aggregate (in percentage terms) is always dampened relative to the individual groups. However, like substitution can actually be amplified relative to the groups.

To see an example where the like substitution effect can overwhelm the unlike substitution effect and create an elasticity greater than either group, imagine a case where we have a price drop in goods 1 and 2 (can be the same or different), a tiny price drop in good 3, and a price increase in good 4. Within groups, changes in shares (normalized by the geometric mean) will be proportional to changes in prices (also normalized by geometric mean of prices), so we will see an increase in the shares of 1 and 2, a small increase in the shares of good 3, and a decrease in the shares of good 4. However, in the aggregate, we instead get an enormous relative change in the amount of good 3 being consumed. Although we do still get less unlike substitution from goods 3 and 4 towards the cheaper goods 1 and 2 (because group B prefers 3 and 4), depending on the weights we could see a large degree of substitution.

Using the Sato-Vartia weights somewhat alleviates this problem as it prevents small price changes from being heavily weighted in the calculation. However, we can tell from the figure that there are still some cases where the elasticity lies above the group elasticities. The shape of the figure helps us understand when this can occur. As the difference in the geometric mean across the sets of similar goods gets small, the unlike substitution effect goes down as well. In other words, there is relatively little movement from goods 1 and 2 to goods 3 and 4 and vice versa. In some cases, the like substitution can be strong enough to push the elasticity above the average of the groups. In this example, this effect is not enough to push the average (shown by the orange line) above 8 conditional on any mean price difference, but this result is dependent on the distribution of price changes as well as the strength of tastes.

Although we cannot say anything in general about the estimated elasticity when we have more than 2 goods, there is still reason to believe that the estimated aggregate elasticity is likely to fall below either of the groups. As discussed above, if the unlike substitution effect is large, the elasticity will always be downward biased. If the prices of goods that share a similar taste structure move together, there will be more pressure to substitute to less preferred goods. In other words, if one group has a strong taste for popcorn products, and another for potato chips, the aggregate consumer will usually have lower elasticity estimates if prices of all popcorn products tend to rise together relative to potato chip products and vice versa. Figure 2.14 shows estimated aggregate sigmas for an example with 200 different products and 10 groups who each have a stronger taste for 10 of the products. In the plot, a correlation of 0 implies that all products change prices completely randomly while a correlation of 1 implies that all products preferred by one of the groups always change prices in the same direction.

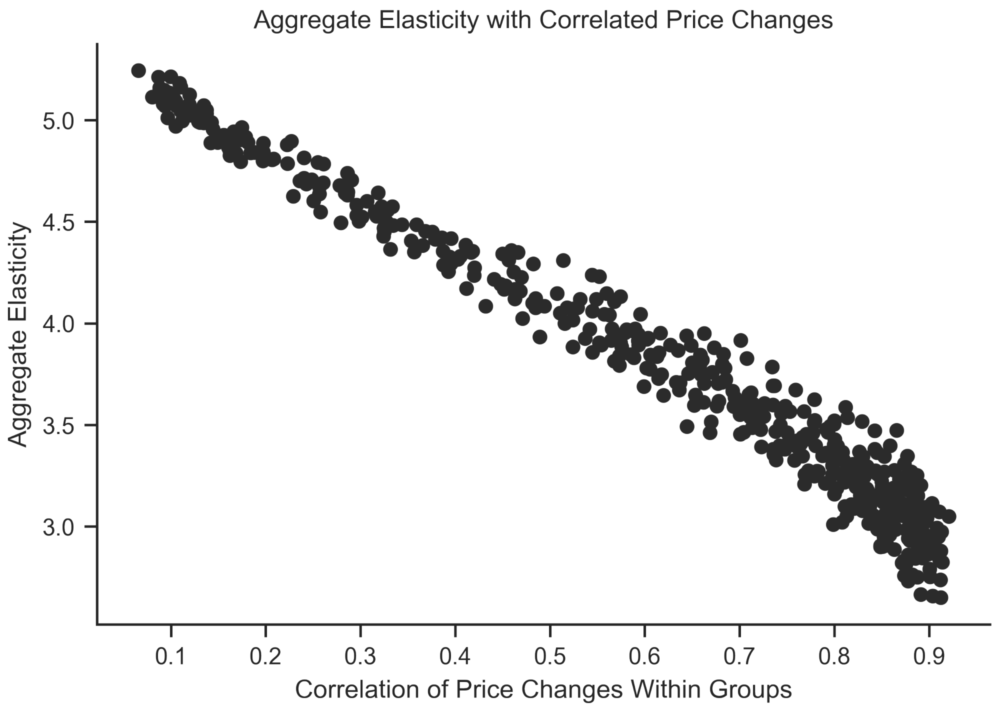

Figure 2.14: Estimated aggregate elasticity against correlation of prices within a common taste set. Results are generated from 10 groups of consumers buying 200 products. Each group has a strong taste ($\phi = 10$) for 10 of the products and $\phi = 1$ for all others. Each group shares a common elasticity $\sigma = 6$. Correlation is generated by adding in a group specific price shock in addition to idiosyncratic price shocks. The relative sizes of these two shocks determines the correlation.

2.7 Appendix II: Estimating Elasticities

The procedure for estimating elasticities throughout the paper relies on the process introduced by Feenstra (1994) and further developed in Broda and Weinstein (2006) and Broda and Weinstein (2010). The method uses log price and share data *doubled differenced*, with the first difference being taken with respect to time and the second difference being taken with respect to the geometric mean. On the demand side, the expression for CES shares logged and doubled differenced can be written

$$\Delta \ln \bar{s}^{*}_{kt} = \beta_0 + \beta_1 \Delta \ln \bar{p}_{kt} + u_{kt},$$

where Δ refers to a time difference and a bar represents a variable normalized by its geometric mean. In other words

$$\Delta \ln \bar{p}_{kt} = \ln \left(\frac{p_{kt}/\tilde{p}_t}{p_{kt-1}/\tilde{p}_{t-1}} \right),$$

where the tilde denotes the geometric mean of the variable. The error term here includes shocks to ϕ. We then set up a simple supply side

$$\Delta \ln \bar{s}^*_{kt} = \delta_0 + \delta_1 \Delta \ln \bar{p}_{kt} + w_{kt},$$

Feenstra (1994) shows that if the error terms from these double differenced equations are orthogonal and heteroskedastic, then the elasticity parameter is identified exactly with two varieties and overidentified with more than two. Broda and Weinstein (2006) show that multiplying the conditions above for each good, weighting the condition for each variety by the amount purchased of that variety and estimating using GMM can recover the elasticity.

2.8 Appendix III

This appendix provides the complete results of running the clustered estimation on each group in the data. For each product group (as classified by Nielsen), the table shows the number of UPCs in that product group (over the entire sample, not all products available in all periods), the number of brands (classified by Nielsen), the number of clusters the DPC algorithm found, and the estimates for elasticity σ and the average annual inflation rate π (as a percent) for both the aggregate and the expenditure weighted mean of the clusters.

Table 2.2: Elasticity and Inflation Estimates for All Product Groups

Product Group	UPCs	Brands	Clusters	Agg σ	Mean σ	Agg π	Mean π
Baking Mixes	11416	1174	10	6.55	6.71	-1.03	-0.57
Baking Supplies	17722	1918	5	6.18	6.66	-0.36	-0.22
Breakfast Food	14163	1134	7	7.27	9.04	-1.03	-0.66
Cereal	26329	3553	3	6.89	7.66	-0.83	-0.61
Coffee	36271	2415	10	7.00	7.78	-1.58	-1.12
Condiments	59980	8288	5	6.72	6.96	-0.36	-0.20
Desserts	11448	1275	15	8.37	8.89	-0.49	-0.26
Flour	3400	714	3	5.37	6.24	0.38	0.43
Fruit - Dried	16778	1617	5	7.21	8.22	-0.96	-0.54
Nuts	37266	2062	8	6.24	7.02	-0.49	-0.25
Packaged Milk	9366	945	10	7.18	8.14	-0.82	-0.42
Pasta	20140	1498	13	6.27	6.94	-0.25	0.08
Pickles, Olives, Relish	22067	2552	9	5.87	7.32	-0.34	-0.00
Dressing, Mayo, Toppings	16127	1851	10	6.87	7.83	-0.60	-0.38
Shortening, Oil	12779	2055	8	6.10	6.84	-0.46	-0.21
Spices, Seasonings, Extract	55685	4369	8	7.46	7.48	-0.22	0.01
Sugar, Sweeteners	4741	724	6	7.32	7.79	-0.32	-0.10
Table Syrups, Molasses	4610	816	2	5.89	7.72	-0.47	-0.10
Tea	24733	2718	8	7.86	7.96	-0.92	-0.79
Vegetables, Grains - Dried	11052	1272	6	7.31	8.13	-0.82	-1.03
Bread and Baked Goods	165378	6927	9	6.64	7.76	-0.70	-0.47
Carbonated Beverages	38141	2808	13	13.27	10.48	-0.24	-0.11
Cookies	62409	5062	5	6.23	7.87	-0.64	-0.28
Crackers	14987	1732	9	7.24	7.71	-0.74	-0.22

Product Group	UPCs	Brands	Clusters	Agg σ	Mean σ	Agg π	Mean π
Snacks	110830	8341	10	7.04	8.30	-1.04	-0.55
Non Carbonated Soft Drinks	29864	4172	8	9.35	8.76	-0.96	-0.78
Baked Goods - Frozen	9657	1229	4	6.93	7.50	-0.24	-0.22
Breakfast Foods - Frozen	8998	844	7	5.89	7.74	-1.00	-0.74
Desserts - Frozen	7818	736	5	6.71	7.13	-0.37	-0.29
Ice Cream Novelties	51963	3084	6	7.27	8.08	-0.46	-0.35
Pizza, Snacks - Frozen	21171	1997	9	7.31	8.50	-1.04	-0.54
Prepared Foods - Frozen	51561	4876	6	7.55	7.99	-0.90	-0.65
Unprepared Meat - Frozen	23973	2808	12	8.82	9.14	-2.18	-1.80
Vegetables - Frozen	22501	1062	13	6.22	7.84	-0.75	-0.40
Butter and Margarine	4232	635	11	4.45	5.65	-0.78	-0.40
Cheese	44823	3260	9	5.64	6.83	-0.78	-0.42
Cottage Cheese/Sour Cream	7766	796	7	4.59	6.51	-0.65	-0.21
Dough Products	4874	377	7	5.66	7.14	-0.02	0.27
Eggs	6168	862	6	4.58	5.15	-0.42	-0.24
Milk	23347	1424	9	7.48	8.23	-0.22	-0.17
Snacks, Spread, Dip (Dairy)	15881	2391	3	7.12	7.48	-0.74	-0.57
Yogurt	20857	819	15	7.68	9.80	-1.24	-1.04
Dressings/Salads - Deli	82322	5601	15	7.67	9.09	-1.98	-1.81
Packaged Meats - Deli	47440	2985	6	5.37	6.25	-1.06	-0.77
Fresh Meat	6021	847	7	7.39	7.35	-0.57	-0.20
Fresh Produce	55756	6444	13	6.86	6.95	-1.36	-1.31
Detergents	18093	1068	12	10.39	9.17	-1.75	-0.93
Fresheners and Deodorizers	40000	2390	6	8.68	9.54	-2.13	-1.28
Household Cleaners	18168	3359	8	6.91	9.19	-1.43	-0.61
Household Supplies	61581	5309	8	7.62	9.56	-0.82	-0.22

Product Group	UPCs	Brands	Clusters	Agg σ	Mean σ	Agg π	Mean π
Laundry Supplies	26808	2326	7	7.97	8.85	-1.29	-0.74
Paper Products	122218	2832	11	9.35	7.99	-0.84	-0.48
Personal Soap and Bath	52084	4654	4	8.69	8.27	-1.39	-1.08
Pet Care	206553	9730	3	8.17	8.74	-1.42	-1.02
Tobacco and Accessories	31867	2965	4	10.16	9.75	0.29	0.51
Wrapping Materials and Bags	21928	1017	11	6.99	7.82	-0.63	-0.15
Beer	31400	17683	2	11.83	12.32	-0.09	-0.14
Liquor	37192	5990	5	11.30	10.07	-0.31	-0.32
Baby Food	8308	425	3	11.98	14.90	-0.96	-0.44
Candy	155005	9919	3	7.48	7.39	-1.00	-0.79
Fruit - Canned	13214	1256	9	6.49	7.57	-0.23	-0.00
Gum	7784	1329	4	7.26	7.80	-2.65	-1.69
Jams, Jellies, and Spreads	21499	2476	10	6.06	6.66	-0.51	-0.19
Juice, Canned, Bottled	50050	3974	10	5.47	6.83	-0.85	-0.49
Pet Food	62421	3829	11	8.15	9.50	-0.98	-0.66
Prepared - Ready to Serve	24155	3188	9	7.06	7.75	-0.26	-0.16
Prepared Foods - Dry Mixes	20997	2214	10	6.21	7.44	-0.97	-0.45
Seafood - Canned	7824	1067	10	6.63	6.89	-0.44	-0.10
Soup	19534	1429	18	6.08	7.03	-0.69	-0.39
Vegetables - Canned	30610	2220	10	5.35	6.34	-0.23	0.09
Automotive	13012	1778	4	7.41	7.38	0.14	0.43
Batteries and Flashlights	20973	1724	11	9.63	7.95	-1.05	-0.45
Charcoal, Logs, Accessories	5725	1051	3	7.68	8.05	-3.32	N/A
Cookware	35087	1907	5	4.61	9.84	-7.41	-2.21
Records and Tapes	375757	5166	2	5.57	6.76	-2.19	N/A

Product Group	UPCs	Brands	Clusters	Agg σ	Mean σ	Agg π	Mean π
Floral, Gardening	8362	1227	6	6.72	7.18	-1.39	-0.48
Party Needs/Novelties	25022	1508	6	5.18	8.58	-1.42	-0.51
Hardware, Tools	68219	4130	7	9.85	9.70	-0.89	-0.27
Housewares, Appliances	73908	6774	3	9.56	7.54	-0.77	-0.56
Pesticides	10570	1380	3	7.19	8.09	-1.17	-0.71
Kitchen Gadgets	169620	8239	7	6.13	6.78	-1.39	-0.52
Shoe Care	3179	313	3	6.75	6.19	-1.93	-1.60
Soft Goods	16692	904	3	8.67	9.68	-0.34	0.06
Stationary, School Supplies	229160	10500	2	8.35	8.26	-0.92	-0.66
Toys and Sporting Goods	15011	699	4	7.76	8.49	-0.93	-0.77
Baby Needs	35361	2473	6	6.58	7.75	-2.18	-0.80
Diet Aids	2663	394	6	6.85	8.06	-0.76	-0.18
Feminine Hygiene	3003	338	4	8.12	8.21	-0.96	-0.38
First Aid	24327	2260	11	8.65	8.90	-1.37	-0.46
Fragrances - Women	56487	4856	8	8.99	9.33	-1.14	-1.04
Hair Care	67066	4420	2	7.13	7.20	-1.70	-1.49
Medications	87962	7222	3	10.13	8.38	-0.61	-0.70
Men's Toiletries	16411	2267	6	9.90	8.68	-1.25	-0.55
Oral Hygiene	26340	2106	9	9.72	9.46	-1.23	-0.82
Sanitary Protection	8480	414	5	8.40	8.95	-0.76	-0.61

CHAPTER 3

Keynesian Dynamics with Customer Markets in a Visual, Interactive, Agent Based Model

3.1 Introduction

The increase in the availability and power of computational methods in economics has predictably led to a parallel increase in the variety of economic models available for researchers to explore. One of the most exciting of these advances has been the rise of agent based models (ABMs). In macroeconomics specifically, development of ABMs has offered an alternative to the dynamic stochastic general equilibrium (DSGE) models that have dominated the profession for decades (see Dawid and Delli Gatti (2018) for a survey of macroeconomic agent based models). While features like heterogeneity and disequilibrium dynamics are often difficult to implement in DSGE models, they become much simpler in computational simulation models that characterize the ABM approach.

This paper contributes to the growing ABM literature by combining the insights of the Keynesian cross with the customer markets model of Phelps and Winter (1970). The combination of these two ideas allows the model to endogenously generate business cycle fluctuations as firms adjust prices and output while navigating a tradeoff between charging a markup and retaining customers. Additionally, I develop a new kind of ABM that emphasizes visibility and interactivity. Using Apple's Swift programming language, I create a standalone program that allows users to adjust the key parameters of the model without needing to change the source code. Once the program is running, users can observe the current state of individual agents in the economy and graphs of aggregate statistics in real time. These features allow users to experiment with the model and ask questions beyond those answered in this paper. A common criticism of ABMs is their "black box" nature. The flexibility afforded to ABMs also represents a challenge for researchers to separate the effects of different mechanisms and determine causality. With dozens or even hundreds of free parameters, ABMs find themselves far from the comparatively rigid structure imposed by the assumptions of DSGE models. Allowing users to observe the mechanics of models in real time could help to reveal how they work.

Previous attempts to bring a deeper level of understanding of ABMs have generally fit into two categories. The first has relied on empirical validation. Windrum et al. (2007) provides an early standard for calibrating parameters and testing the results of an ABM against real world data while Fagiolo et al. (2017) gives a more recent update. Various methods have been proposed to either infer parameter values and agent behaviors from micro data or to search over the entire possible parameter space to best match the results to the data. However, even though data can potentially offer a useful constraint on ABMs, not all models should be required to perfectly or even approximately match the data. As Kydland and Prescott argued in their early defense of real business cycle methodology (Kydland and Prescott, 1991), there is little reason to believe that stylized models should ever produce results resembling reality.

An alternative approach offers a more theoretical solution by using phase diagrams to demonstrate the effects of changing parameter values (Gualdi et al., 2015). This method allows the researcher to show the effects of a wide range of model specifications without taking a stand on the most appropriate parameter values. A potential problem with phase diagrams comes from the difficulty of capturing interactions between variables without resorting to complicated multidimensional diagrams.

More importantly, neither of these solutions fully opens the black box of an agent based model. Empirical validation methods certainly add some transparency to modeling choices, but even a perfectly specified ABM can produce seemingly mysterious results. Emergent behavior, the trademark feature of many ABMs and one of the reasons they are so appealing as a modeling technique, also ends up making their implications difficult to parse. With thousands or tens of thousands of individual agents making individual decisions, observing aggregate results does not always help a researcher understand the mechanisms driving the results. By producing charts or tables describing the distribution of agents, a modeler can somewhat mitigate this issue, but static snapshots can only offer a partial solution.

As technology and the tools offered to researchers continue to improve, so too should the media researchers use to convey their findings to readers. One of the primary goals of this paper is to take a step toward better communication of agent based models in macroeconomics. It works toward this goal in two ways. First, it offers a complete graphical representation of the model that can be observed in real time as agents interact. Both individual agents as well as aggregates and averages for the entire economy can be observed and graphed dynamically so that users can see not only the end result of a model run, but also the inner workings of the economy. In this sense the model is similar to ABMs created in NetLogo (Tisue and Wilensky, 2004) or other graphically focused programs. In macroeconomics, the Java Agent Based Macroeconomic Laboratory (JAMEL) model of Seppecher (2012) also offers a program that produces graphs and statistics in real time as the simulation runs (applications of the model can be found in Seppecher and Salle (2015) and Seppecher et al. (2018)) However, the model presented here offers a richer visual and interactive component.

The second key component of the model is complete flexibility for users to change parameter values. Before each run, users can choose the number of agents, as well as behavior regarding production functions, adjustment behavior and policy. While a benchmark calibration is provided later in this paper, users are free to observe how changing parameter values can affect the economy. Combining this option with the graphical interface described above aims to make the effects of changes as clear as possible. An ABM can never hope to match the clarity of a general equilibrium model with an analytic solution, but seeing the results of a simulation as it runs offers its own perks, allowing for the actions of individual agents to be directly observable over time.

Of course, the visual and interactive aspects of the model are only useful if the model itself can say something interesting. Although the model described in the paper does not strive to give quantitatively accurate predictions, it does try to simulate realistic economic interactions on a small scale. It is designed to be a toy economy, a playground to test ideas and observe outcomes under different specifications. For now, the setup of the model is too simple to allow for comparisons to data or any kind of estimation, but the framework is flexible enough that more realistic additions could be added in future work.

Inspiration for the model's features comes from two sources. At its core, the model is an attempt to move classic Keynesian cross intuition into a new setting. Production decisions are based on expected demand and consumers spend based on the income they receive. Unlike static Keynesian models, the economy is constantly in a state of disequilibrium as firms fluctuate around a desired level of inventories. Increases in demand run down a firm's inventories, causing them to increase production and employment. In this sense, the economy conforms to the simple Keynesian multiplier intuition. Augmenting this setup is a production process where consumption firms are required to hire workers and purchase machines from investment firms in order to produce.

The other guiding principle of the model derives from the literature on customer markets formalized by Phelps and Winter (1970). In customer market models, firms have a base of customers that only gradually changes according to firm pricing decisions - firms with higher prices slowly lose customers to cheaper competitors. Empirical evidence has demonstrated that attempts to retain or attract new customers is one of the main considerations that goes into firms pricing decisions. Within more standard economic theory, customer market models have been given some recent attention (Gourio and Rudanko, 2014; Paciello et al., 2016), but they haven't been focused on as much in agent based setups.

In the macroeconomic ABM literature, the model described in this paper aligns most closely with the Keynes-Schumpeter evolutionary model first described in Dosi et al. (2006) and expanded in later papers (Dosi et al., 2010, 2013, 2015, 2017). Relative to that framework, this paper adds a visual and interactive component as well as a different pricing mechanism driven by customer markets. The focus on customer markets is also in the spirit of Howitt and Clower (2000) (and later work by Ashraf et al. (2016, 2017)), although their implementation of stores and trading relationships centers around matching consumer tastes to heterogeneous while this paper focuses on the competition between firms producing similar products. The previously mentioned JAMEL model also relates to this work, but its focus on the money market offers a different perspective on an agent based Keynesian model.

With the exception of the JAMEL model, none of the macroeconomic ABMs described above provide a visual component, and none make it as easy as the program described in this paper to adjust parameter values or see responses to different policies in an interactive setting. This foundation hopes to provide researchers with a new toolbox to explore important macroeconomic questions.

3.2 Model

The model is centered around four different kinds of agents: consumers, consumption firms, investment firms, and the government. Consumers purchase a single homogeneous consumption good from consumption firms, who require machines from investment firms to produce. Prices are governed by competition in customer markets. Each consumer can purchase from only one firm in each period (and each consumption firm can purchase from only one investment firm) and firms with above average prices will see customers leave to search for a better price.

The basic sequence of events of the economy is described below

1. Consumers make decisions on how much of their wealth to spend
2. Workers decide whether to quit or remain in their job
3. Consumption and investment firms decide how much to produce based on past sales and current inventory levels
4. If desired production is different from feasible production, firms hire/fire workers (and consumption firms buy machines)

5. Firms sell their product at their current price. If sales differ from expected sales, the remainder is captured by a change in inventories.

6. Workers are paid the wage rate of their current employer

7. Firms set prices and wages for next period

8. Customers decide whether to remain at their current firm or leave

9. Machines depreciate

10. Government makes spending and monetary policy decisions

The following subsections describe each of these procedures in more detail.

3.2.1 Consumers

The economy is populated by a fixed number n_c of consumers. Consumer spending behavior is Keynesian in nature in the sense that consumption and saving decisions are driven by current disposable income and marginal propensity to consume. In the standard setup, MPC is fixed over time for each consumer, although it can vary across consumers. Because consumption decisions are made before workers are paid, I assume that all consumers enter the economy with cash holdings h_0 in period 0 (this is the only source of money when the simulation begins so the sum of consumer cash holdings is also equal to the total money supply in period 0). Consumers have a desired level of cash holdings. When savings pushes cash holdings above this desired level, consumers add the excess to their consumption spending. The spending decision for a consumer can therefore be described as:

$$c_{st} = by_{dt} + \varepsilon(h_t - h^*), \tag{3.1}$$

where c_s is a consumers consumption spending, b is their marginal propensity to consume, y_d is a consumer's disposable (after tax) income, h is cash holding, h^* is desired cash holding, ε is a parameter between 0 and 1 that determines how much of their excess cash they spend on consumption each period, and t is a time subscript.

In order to encompass consumption smoothing behavior, consumers with no income (unemployed workers) attempt to keep their consumption spending constant by spending down their cash holdings when they are unemployed (note that they do not consider real consumption here). Therefore, for an unemployed worker we have

$$c_{st} = c_{st-1}. \tag{3.2}$$

Saving for each consumer can then be taken out as a residual

$$s_t = y_{dt} - c_{st}. \tag{3.3}$$

Consumers purchase consumption goods from a single firm each period. At time 0, they are randomly assigned to a consumption firm and will continue to purchase goods from that firm unless they decide to leave and search for another firm (a decision described in more detail below). Real consumption for an individual can then be defined total consumption spending divided by the current price at the consumer's current firm

$$c_{rt} = c_{st}/p_{ct}. \tag{3.4}$$

where c_r refers to real consumption and p_c is the price at the consumer's current firm.

Labor is supplied inelastically by consumers and there is no concept of leisure in the model. In period 0, consumers are randomly assigned to an employer. They are paid a wage rate specific to their employer at the end of each period (after production). They also pay taxes to the government so that their disposable income is

$$y_{dt} = w_{ct} - T(w). \tag{3.5}$$

where w_{ct} is the wage rate of the consumer's current employer and $T(w)$ is taxes paid to the government (which can depend on the consumer's income).

Although consumption spending is so far quite mechanical, consumers adapt to the state of the economy in two important ways. First, workers quit their job with some probability.

The probability of a worker quitting a firm with wage w_c is given by

$$
\begin{aligned}
P_q &= \frac{1}{\pi}\arctan\left[\left(\frac{\tilde{l}-\tilde{w}}{\theta}+\frac{\pi}{2}\right)\right]\\
\tilde{w} &= 100\frac{w_c-\bar{w}}{w_c}\\
\tilde{l} &= \frac{n_c}{n_c-L}.
\end{aligned}
\tag{3.6}
$$

where $\tilde{w}$ is the percentage deviation of the firm's wage compared to the average wage $\bar{w}$ across all firms (in both sectors), $\tilde{l}$ is a measure of how far the current employment level L is from full employment ($n_c = L$), and θ is a parameter that can be thought of as a kind of labor market tightness (as theta increases, effects of wage differences on quit rates diminish). Intuitively, when the wage of an employer is low relative to the average, or when the economy gets closer to full employment, the probability of an employee quitting increases. The exact functional form for the quit probability was chosen because it was an increasing function (in $\tilde{l}-\tilde{w}$) bounded by 0 and 1. Other functions with this property should work in theory, although experiments with logistic functions found that it produced large fluctuations as it approached the full employment limit.

Customers make similar adjustments in the product market. Here I split the market into firms with above average and below average prices. Firms with high prices lose customers who enter a pool of searching consumers until they are paired to a lower priced firm. For a firm with a price p_c above the average price, the probability of losing a customer is given by

$$
\begin{aligned}
P_{lc} &= 1-e^{-\xi_l\tilde{p}}\\
\tilde{p} &= 100\frac{p_c-\bar{p}}{p_c}.
\end{aligned}
\tag{3.7}
$$

where $\tilde{p}$ is the percentage deviation in price from the average and ξ_l is a parameter controlling the speed of customer loss.

For a firm with a below average price, the probability of gaining a customer is

$$
\begin{aligned}
P_{gc} &= 1-e^{\xi_g\tilde{p}\tilde{s}}\\
\tilde{p} &= 100\frac{\bar{p}-p_c}{p_c}\\
\tilde{s} &= \frac{n_s}{n_s-n_c}.
\end{aligned}
\tag{3.8}
$$

where $\tilde{s}$ is an adjustment to the probability that depends on the current number of searching consumers n_s relative to the total. These two equations imply that firms with high prices will lose consumers over time while firms with low prices will gain them. We will see that this behavior drives price competition among firms that does not lead to monopoly prices (as in the "Diamond paradox" of many mathematical product search papers stemming from Diamond (1971)), or to Bertrand marginal cost pricing. In this sense, the mechanism is similar to Phelps and Winter (1970), but the agent based setup allows for the exploration of disequilibrium dynamics unlike the symmetric equilibrium studied in that paper.

3.2.2 Firms

As explained above, there are two types of firms in the economy. Consumption firms produce a single homogeneous good for sale to consumers, while investment firms produce machines that are required for production of the consumption good.

3.2.2.1 Consumption Firms

More specifically, the production process for consumption firms takes capital (machines) and labor as complementary. The maximum production for a consumption firm is given by

$$y_{max} = \Big(\min(zl, \gamma m)\Big)^{\alpha_c}. \tag{3.9}$$

where l is the number of workers hired by the firm, z is the productivity of the firm, m is the number of machines they currently hold, γ is the capacity of each machine, and α_c controls the returns to scale of the production process. In order to prevent all production being carried out by a single firm, returns to scale are restricted to be decreasing. A firm can always choose to produce less than its maximum capacity, but can only produce more if it hires more labor or purchases more machines.

Each period, a consumption firm decides how much to produce based on its expected sales as well as the current level of inventories. Firms want to hold a fixed level of inventories to be prepared for unexpected changes in demand. Planned production can be written formally as

$$y_{plan} = \max\Big(E(d_t) - (x_t - x_c^*), 0\Big). \tag{3.10}$$

where d_t is demand for the firm's goods at time t, x_t is the firms current inventory holding, and x_c^* is their desired level of inventories. The max operator serves to ensure that the firm never wants to produce a negative quantity. When inventories expand, the firm may stop production and let their inventories run down, but they never have any incentive to destroy existing goods. For this paper, I assume that expected demand is simply demand from the previous period (i.e. $E(d_t) = d_{t-1}$). Other expectation systems (longer lags, etc.) could be easily implemented.

Given this production plan, the firm then calculates how many production units (the minimum of workers and machines - both measured in efficiency units) it would need to carry out its plan. A worker carries in the workers it had hired from the previous period and the machines it had purchased. Given these resources, if planned production is greater than maximum production, it attempts to hire additional workers to the point where production becomes feasible. However, attempts to hire are not always successful. The probability a hiring attempt ends up being successful depends on the current wage offered by the firm as well as current labor market conditions. I choose a functional form matching that of equation 3.6 governing the probability of a worker quitting. Specifically, a hiring attempt will be successful with probability $P_h = 1 - P_q$. Once again, this form implies that firms that offer higher wages will have an easier time finding new workers than those that offer lower wages and that hiring gets harder as unemployment gets closer to 0.

Due to this effect, firms adjust wages in order to be competitive in the labor market. When a hiring attempt is unsuccessful, a firm responds by increasing its wage by a fixed percentage. For simplicity, in its current form the model has no mechanism for downward nominal wage movements, which corresponds to the empirical literature that tends to find nominal wages are sticky downwards (Barattieri et al., 2014), but other wage adjustments could be easily added in extensions to the model.

On the other hand, if the firm has more workers than it needs for its current level of production, it considers whether to fire some workers. Due to the uncertainty in the ability to hire workers whenever it wants, I assume that firms try to avoid firing workers when there is a single bad period. Instead, firms look at the last few periods and only fire workers if they had too many workers in every one of those periods.

After hiring or firing workers, the firm once again calculates its maximum production capacity. If the number of machines it currently owns is less than is required for its production plan, it buys new machines from its current supplier (supplier choice will be discussed in section 3.2.2.2). Machines cannot be destroyed and they remain in a firm's possession after use. However, each period, each machine (whether used in production or idle), ages by one period. After a fixed number of periods, depreciation renders the machine unusable and the firm needs to buy a new machine to replace it.

With the necessary adjustments completed, each consumption firm then produces goods according to its plan and sells to meet the demand of each of its customers (under the assumptions about demand discussed in section 3.2.1). As mentioned earlier, if sales differ from expectations, the difference is taken from inventories (it is possible inventories drop to zero, in which case the firm only sells its current production, but this scenario should not occur under reasonable parameter specifications). The firm adds the revenue it receives to its current cash holdings and then pays its workers out of its cash holdings. I assume that firms can have negative cash holdings, which essentially means the model includes costless borrowing from the government. In future work I hope to relax this assumption by adding a functioning bond market, but this paper does not include this feature.

A firm's profit is calculated by subtracting labor cost and the cost of machines from revenue. We can write this explicitly as

$$\pi = \sum_{i=1}^{n_c} p_t c_{it} - l_t w_t - \frac{m_t}{\Omega} p_{mt}. \tag{3.11}$$

where n_c is the current number of customers at the firm, p_t is the price of their good, c is consumption, l is the number of workers the firm employs, m_t is the number of machines the firm currently holds, and $p_m t$ is the price of a machine from their current supplier. Machine price is divided by its age Ω so that the cost of a machine is spread out across its lifetime.

Following the customer adjustment process described in section 3.2.1, each consumption firm constantly adjusts its price in order to attract customers. It faces a tradeoff in price setting. With two high a price, a firm will lose all of its customers. Too low, and it won't be able to make a profit. To balance these costs and benefits, I use a simple set of three decision rules.

1. If a firm has zero customers and positive inventories, reduce price
2. If a firm has negative profit, or more customers than average and less profit than average, increase price
3. If a firm has fewer customers than average, and positive profit, reduce price

(Note: firms can never have negative prices - and will never want to under standard configurations)

These three rules capture the idea that firms can only partially extract surplus from their customer base. A firm only tries to increase its price above its costs when its customer stock is sufficiently large. Again, this setup is designed to match the empirical findings of Blinder et al. (1998) and others (e.g. Greenslade and Parker (2012)) that show that customer retention factors heavily into a firm's decision to change its price. The three rules intentionally exclude more complicated pricing decisions involving forward looking behavior and explicit profit maximization. Instead, they fall more in line with Herbert Simon's concept of "satisficing" (Simon, 1956). These rules imply that each firm has some idea of the average profitability and popularity (in terms of number of customers) of other firms in the industry, but they do not need knowledge of the exact quantity of labor and capital necessary to maximize profit as in standard neoclassical models.

To summarize the timeline for consumption firms, the sequence of events proceeds as:

1. Calculate maximum production given current resources (equation 3.9)

2. Calculate planned production based on past sales and current inventories (equation 3.10)

3. If planned differs from maximum production, hire or fire workers and buy machines accordingly

4. Sell to current customer stock. Discrepancies between planned and actual sales come out of inventories

5. Pay workers and determine total profit (equation 3.11)

6. Adjust price based on number of customers and current profit

7. Gain or lose customers based on equations 3.7 and 3.8

3.2.2.2 Investment Firms

Investment firms produce machines in order to sell to consumption firms. They hire from the same pool of workers as consumption firms and therefore face the same probability of a working quitting (equation 3.6) and of being able to hire a new worker. The production process to produce machines only requires labor and works a bit differently than production of consumption goods. The most important difference in the two production processes is that machines require time to build. Economists have long studied the effects of differences in time of production processes dating back to at least Hayek (1932) and agent based models offer a tractable way to explore these effects.

When a new machine is created, it is put into the investment firm's pool of "machines in progress." An investment firm will then try to advance each machine in its pool one period closer to completion using workers. The number of machines a given number of workers can advance is represented by

$$l_e = l^{\alpha_i}. \tag{3.12}$$

where l_e can be thought of as effective units of labor after taking into account the returns to scale α_i

Each investment firm decides how many machines to produce similarly to the decision process of consumption firms. In this case, we have

$$m_{new} = E(m_c) - (x_t - x_i^*) - m_p. \tag{3.13}$$

where m_{new} is new machines added to the production queue, m_c is machines purchased by consumption firms, x_t is current machine inventories, x_i^* is desired machine inventories, and m_p is machines currently in progress. Due to the nature of investment decisions, investment decisions are lumpy and include zeros in some periods (matching basic stylized empirical facts about investment behavior - Doms and Dunne (1998)). Therefore, m_c is estimated by averaging across a number of periods. In the standard setup, I set the number of consideration periods equal to the time to build a machine. Demand for investment goods comes from consumption firms and each investment firm sells to a stock of customers that adjusts over time.

As with consumption firms, given the planned production, an investment firm then calculates the number of workers it needs to carry out production and hires or fires workers as needed. Hiring and firing mechanics work exactly as in consumption firms. Pricing decisions also work similarly, following the same three rules as consumption firms. One slight difference is that investment firms consider profits over a period rather than static profits again in order to avoid excessive volatility in prices. Static profit is

$$\pi = \sum_{i=1}^{n_c} p_t m_{it} - l_t w_t, \tag{3.14}$$

which is just the sum of demands for machines for all of its customers minus its labor cost. Profit for the consideration period is then found by summing static profits.

Customer dynamics also operate similarly. If an investment firm charges a price above the average over all investment firms, it has a chance to lose a customer (remember that customers here are consumption firms). Unlike in the consumption product market, there is no pool of searching consumption firms. Every consumption firm is always matched to a single supplier. With some probability, a consumption firm customer at each investment firm will compare the price it currently pays to that of one other investment firm. If the price is lower, it changes suppliers. Otherwise it stays. The functional form for the probability is similar to that of a consumer leaving its current store

$$\begin{aligned} P_{li} &= 1 - e^{-\xi_{li}\tilde{p}} \\ \tilde{p} &= 100\frac{p_i - \bar{p}}{p_i}. \end{aligned} \tag{3.15}$$

where once again the probability depends on the investment firm's price p_i compared to the average price $\bar{p}$ and a parameter ξ_{li}. To reiterate, this probability only governs the probability that a consumption firm tries to compare prices. They only leave if the new firm they sample actually has a lower price. This distinction means that consumption firm-supplier relationships are harder to break than customer-consumption firm ones, which seems like a plausible assumption.

To summarize the sequence of events for investment firms

1. Determine number of new machines to build (based on equation 3.13)
2. Hire/fire workers corresponding to current production level (new machines + current stock in progress)
3. Sell machines to current customers as demanded
4. Pay workers and determine profit (3.14)
5. Adjust price
6. Gain/lose customers based on equation 3.15

3.2.3 Government

In this version of the model, the government plays a relatively passive role. The model is stock flow consistent, which, combined with the fact that there is only one currency, no bond market, and the model is a closed economy, makes monetary and fiscal policy essentially interchangeable. Government spending can be thought of as creating treasury bonds that are instantly purchased by the central bank, adding to the money supply. On the other hand, taxes draw money out of the money supply.

In the current form of the model, the government has three main functions. First, it charges an income tax on consumers. Since wages are the only form of income, the tax is also equivalent to a tax on wages. With the tax, equation 3.5 becomes

$$y_{dt} = w_{ct}(1 - \tau), \tag{3.16}$$

where τ is the income tax rate. Government revenue is therefore,

$$G_R = \sum_{i=1}^{n_e} \tau w_{it}. \tag{3.17}$$

where n_e is the number of employed workers throughout the economy and w_i is the wage of each worker (which may be different depending on their current employer).

Though the model can easily allow for government spending of many kinds, the example used in this paper is unemployment benefits. Whenever a worker is unemployed they receive some percentage of the average wage. In the current model, there are no other sources of government spending so that total government expenditure is given by

$$G_E = \zeta n_u \bar{w}, \tag{3.18}$$

where ζ is the percentage of the average wage the government pays out, n_u is the number of unemployed workers, and $\bar{w}$ is the average wage. The government deficit/surplus can then be calculated by subtracting equations 3.17 and 3.18

The final job of the government is to handle monetary policy for the economy. Again, any dollar that is spent by the government automatically add to the money supply while taxes are removed from the money supply. In the model, government debt is actually irrelevant to the functioning of the economy since there are no bonds and it is never expected to be repaid. "Borrowing" in this case is simply getting the central bank to "print" money. Without a bond market or banks, traditional monetary policy does not have a place in the model. However, the government can choose to increase the money supply through "helicopter drops" to consumers.

Currently, monetary policy in the model is conducted through simple monetary base targeting. The target level for the base increases at a fixed rate and if the actual money supply falls below target, the central bank increases the money supply by printing money and adding it to each consumer's cash holdings. The purpose of this simple policy is mainly to prevent the money supply from falling to zero. Without it, if the government runs persistent budget surpluses, the total cash holding in the economy will eventually fall to zero and spending collapses. As with government spending, the model is flexible enough to incorporate various types of monetary policy rules (inflation targeting, nominal GDP targeting, etc.). Alternatives will be discussed in section 3.5

3.2.4 Aggregation

As an agent based model, aggregate variables are calculated by summing over individual agents. For example, aggregate consumption can be calculating by summing the individual consumption of individual agents. However, some aggregate variables are not trivial to aggregate. It is unclear, for example, how real GDP should be calculated in the model. While real consumption can be obtained simply by adding up the production of each consumption firm and real investment is similarly the sum of investment firm production, summing real consumption and investment would be attempting to add goods with entirely different units. One unit of consumption is in no way comparable to one machine. Calculating real GDP would therefore require the use of some kind of price index (a model version of the GDP deflator used in actual aggregate accounting). Rather than attempt this exercise, I instead refrain from discussing aggregate output and instead focus on consumption and investment sectors in isolation whenever talking about real production.

Even nominal quantities present some challenges. Using the expenditure approach to try to calculate nominal GDP runs into the issue of how to measure inventory adjustments, a topic that remains under some debate (Reinsdorf, 2007). It is clear that, in theory, production that is added to inventories should still be counted in that periods aggregate output. Less clear is the correct price to use when adding these quantities. For example, imagine that a firm sets an exceptionally high price, which causes demand to fall and therefore much of the firm's production ends up in inventories. Over time, the firm reduces its price and eventually sells all of its inventories. Using the initial high price to calculate nominal GDP doesn't give an accurate picture of production since those goods would have never sold at that price. A perfect measure would observe the sale price of each individual unit and update nominal GDP in the time period that it was produced. However, this kind of measurement is both impractical and unhelpful if the goal is to observe values in real time.

Therefore, despite the possible flaws in accounting methodology, the model calculates nominal GDP simply by multiplying current production by the current price of the firm that creates it. Changes in inventories for consumption firms are calculated as the residual of total production by consumption firms and consumer demand for that output. This calculation then allows total investment to be constructed as production by investment firms (new machines) and changes in inventories of both consumption and investment firms.

3.3 Simulating the Model in a Visual Interactive Environment

The model above aims to provide a reasonable foundation upon which to build, but it does not pretend to be a major contribution to our understanding of economic phenomena. The more important contribution of this paper is the ability to run the model described above in a visual, interactive environment. In this section I explain how the program works.

3.3.1 Overview

Figure 3.1 shows the main screen of the program during a model simulation. At any moment during the simulation, the program displays every single agent that makes up the model. The four different types of agents are each identified by a square of a different color. Consumers are blue, consumption firms red, investment firms yellow, and the government is brown. As the economy runs, consumers who become unemployed and firms who have no customers change to black. When a consumer is employed or a firm with no customers attracts a new customer, their color is restored.

Stats for the aggregate economy are displayed on the left in green text. Displayed variables include nominal GDP, real consumption, average profit (of consumption firms), average price of the consumption goods, the number of consumers searching for a new store, the average machine price, the unemployment rate, and the average wage. As the model runs, these variables are updated in real time

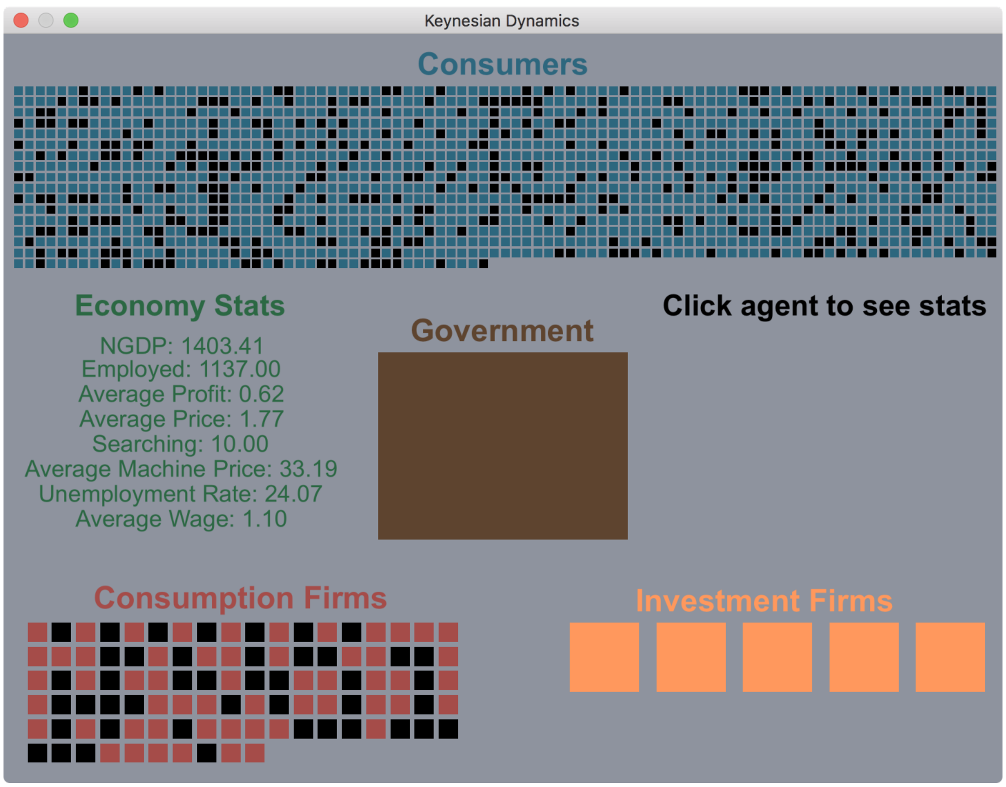

Figure 3.1: The main screen during a model simulation

3.3.2 Interaction

As the economy runs, it allows the user to interact and observe how changes in the economy affect individual agents. Figure 3.2 shows how the program reacts when each type of agent is selected. By clicking an agent, the right panel changes to show the individual stats for that agent. It also changes the color of agents to reflect that agent's current relationship with other agents. When an agent is clicked (or hovered over with the mouse), its color changes to yellow.

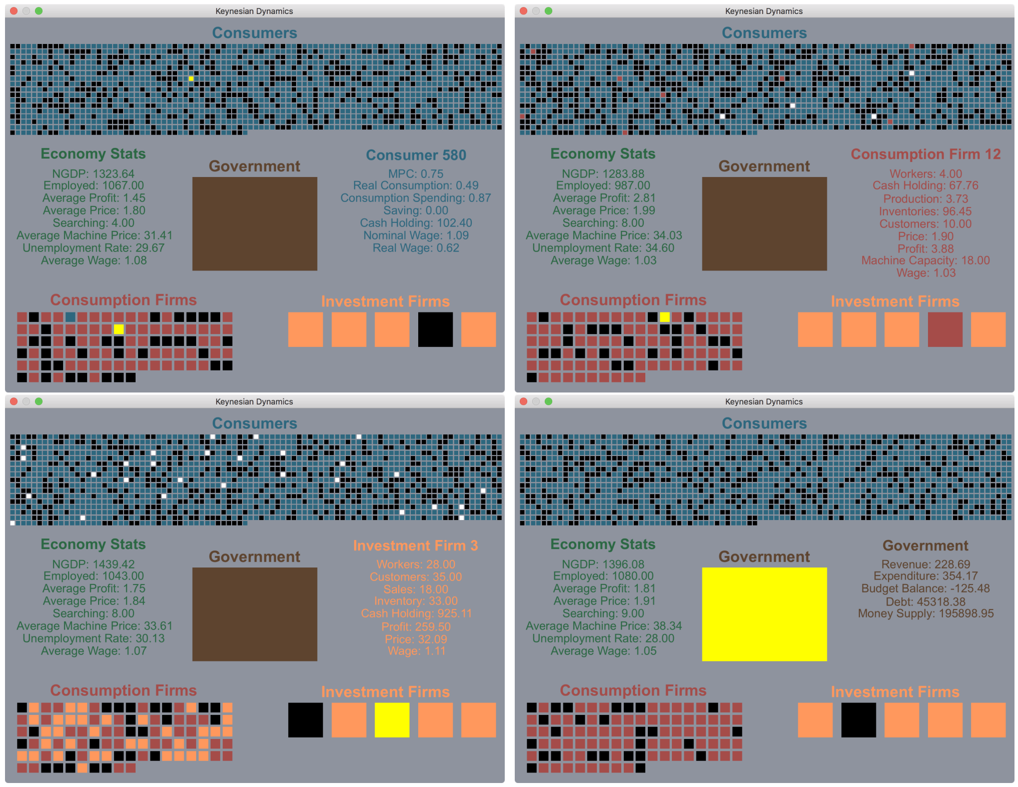

Figure 3.2: Examples of program output when an agent is selected. From top left to bottom right, the figure shows a consumer, a consumption firm, an investment firm, and the government being highlighted

Clicking a consumer (top left panel of figure 3.2) shows its marginal propensity to consume, current real and nominal consumption, its saving, cash holding, and nominal and real wage. It also highlights the consumption firm that it is currently buying goods from (that firm will change color to blue), and its current employer (could be a consumption or investment firm - which will turn yellow).

A clicked consumption firm (top right panel of figure 3.2 will show how many workers it currently employs, its cash holding, its current level of production, its inventories, the number of customers currently buying from it, its price, profit, machine capacity, and wage. It also highlights all of the workers it employs (consumers it employs turn white), and its customers (turn red).

Similarly, an investment firm (bottom left panel of figure 3.2) shows the number of workers and customers, sales, machine inventories, cash holding, profit (period profit as described in section 3.2.2.2), current machine price, and current wage. It also reveals all of the investment firms customers in orange and consumers it employs in white.

Finally, hovering over or clicking on the government will show its current revenue, expenditure, budget balance (surplus or deficit), debt, and the current money supply of the economy. Since government benefits in this version of the model are only paid to unemployed workers, there is no color coding given for recipients (they are just all of the unemployed consumers), this feature could easily be added in future versions with more complicated government spending programs.

When any stat is clicked, a graph will be displayed in the center of the application. The graph shows the evolution of whichever variable is clicked over time. Figure 3.3 shows an example of a graph running in the program. Time runs along the horizontal axis and the variable chosen (in this case nominal GDP) on the vertical. As the economy continues to run, the graph updates in real time. The graph continues to plot new points starting at the period, t, when a variable is clicked and will plot up to 500 additional time periods of data. After 500, the graph will continue to update with new data, but will drop the earliest points so that it always shows 500 total datapoints (the number of time periods can be configured within the program).

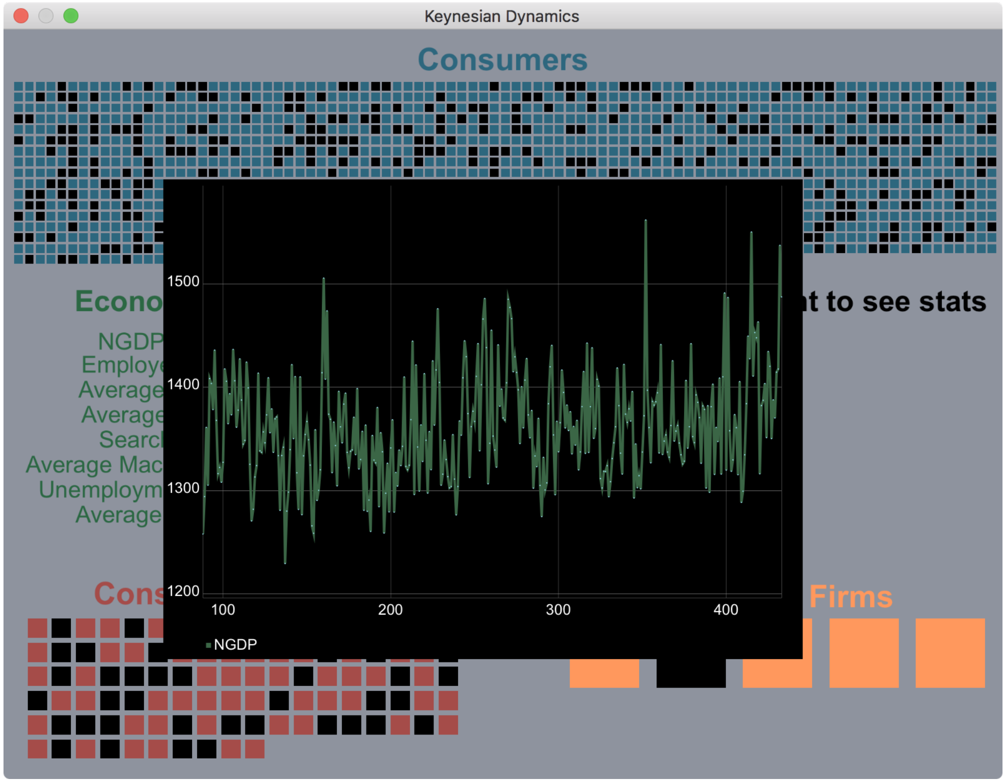

Figure 3.3: An example of a graph running in the program

3.3.3 Changing Parameters

In addition to the interaction described above, the other key feature of the model is the ability to change the key parameters of the model. Figure 3.4 shows the parameter selection screen in the program. Parameters that can be changed are divided into 6 categories. First, parameters that affect the entire economy are the number of consumers (n_c), the number of consumption firms (n_{cf}), the number of investment firms (n_{if}), and labor market tightness (θ).

For a consumer, parameters include the marginal propensity to consume (b) (which can further be configured to be uniform across consumers or randomly distributed - at the moment, uniform distribution is the only option), the excess cash spending rate (ε), and their desired cash holding (h^*).

Consumption firm parameters are their desired inventory (x_c^*), the parameter governing the extent of decreasing returns to scale (α_c), the customer loss rate (ξ_l) and the customer gain rate (ξ_g).

Investment firm parameters are the capacity of each machine (γ), the age at which a machine depreciates (δ), the time it takes to build a machine (χ), desired machine inventories (x_i^*), the parameter governing the extent of decreasing returns to scale (α_c), and the customer loss rate (ξ_{li}).

Government parameters are the income tax rate τ), the unemployment benefits rate (ζ), and the growth rate of the money supply target (g_m).

Finally, the number of periods for the graphs to show in the program can also be adjusted on the setup screen.

Note that some parameter values cannot be accepted (for example negative values). Attempting to input an invalid parameter value will lead to the program automatically defaulting to a parameter within the acceptable range.

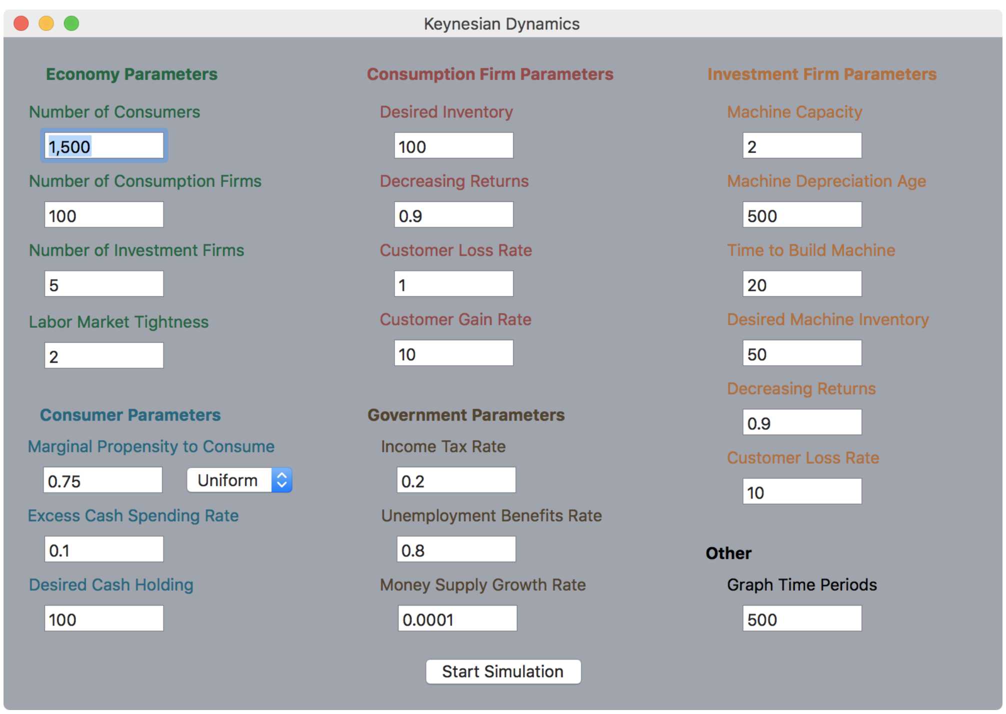

Figure 3.4: The parameter selection screen

3.4 Benchmark Calibration Results

This section offers an example calibration of the model in order to demonstrate some basic results. However, as discussed above, one of the primary motivations of the design of the model is to move away from allowing rigid parameterizations to drive the result. The utility of this kind of model should be in allowing anyone to play with different parameterizations and explore the results themselves. Still, as the creator of the model, I can share some of the results I have found and explain why I chose the benchmark calibration I have chosen. Table 3.1 offers a list of the benchmark parameters (these can also be seen in figure 3.3.3 or by running the program and seeing the default values).

In total, there are 20 parameters in the model that require calibration. The large number of free parameters obviously gives the model flexibility in terms of the results it can potentially generate. This flexibility can be both a blessing and a curse because identifying such a model and estimating it quantitatively becomes extremely challenging if not impossible. The model is better suited for qualitative results, for testing simple theories of how the economy should work by adjusting parameters in order to see if they match intuition. In this sense, the model aims to act as a testing ground for new ideas that can then be further developed into quantitative experiments with more complicated models.

Two important ideas inform the design of the model. First, it should be able to generate fluctuations without relying on exogenous shocks. Accomplishing this goal is much easier in an ABM than in DSGE alternatives, which often rely on questionably sourced TFP shocks to generate realistic looking fluctuations. In contrast, agent based models can more easily produce chaotic behavior that is entirely deterministic but still incredibly hard for agents in the model to predict (see for example Brock and Hommes (1997)). This unpredictability gives deterministic models an answer to the Lucas Critique. The usual justifications for including rational expectations, that smart agents will eventually learn the model and begin to use forward looking predictions, becomes much less appealing in the face of unpredictable chaos.

Table 3.1: Benchmark Calibration

Parameter	Description	Benchmark Value
n_c	Number of Consumers	1500
n_{cf}	Number of Consumption Firms	100
n_{if}	Number of Investment Firms	5
θ	Labor Market Tightness	2
b	Marginal Propensity to Consume	0.75
ε	Excess Cash Spending Rate	0.1
h^*	Desired Cash Holding	100
x_c^*	Consumption Firms Desired Inventory	100
α_c	Decreasing Returns for Consumption Firms	0.9
ξ_l	Consumption Firm Customer Loss Rate	1
ξ_g	Consumption Firm Customer Gain Rate	10
γ	Machine Capacity	2
δ	Machine Depreciation Age	500
χ	Time to Build a Machine	20
x_i^*	Desired Machine Inventory	50
α_i	Decreasing Returns for Investment Firms	0.9
ξ_{li}	Investment Firm Customer Loss Rate	10
τ	Income Tax Rate	0.2
ζ	Unemployment Benefits	0.8
g_m	Money Supply Target Growth Rate	0

The second feature the model hopes to encompass is the ability to remain relatively stable over time. Unlike standard DSGE style models, the model in this paper does not explicitly define a steady state or have any built in reason that it should converge to one. However, it does contain elements that serve to stabilize many of the key variables. For example, the price adjustment mechanism described in section 3.2.2.1 causes firms to hover around 0 profit as they compete to retain customers (and increase prices when profits get too low). This mechanism also maintains a relatively stable price level (around 2 in the benchmark calibration with no money supply growth).

These two characteristics of the model lead to nice looking economic fluctuations around a somewhat stable mean. To illustrate this property, figure 3.5 shows a sample simulation of nominal GDP using the benchmark calibration described in Table 3.1. The gray lines in the graph depict the actual series for NGDP while the black line shows a 20 period moving average. It is important to keep in mind that the model does not necessarily have a natural mapping to real GDP data. The adjustment procedures that form the backbone of the model are probably more conducive to something like a weekly frequency. The volatility of the series is therefore much higher than what would be seen in the GDP data. Despite this qualification, the series exhibits both of the desired properties mentioned above. Nominal GDP fluctuates around a mean of about 1650, peaking at around 2000 and bottoming out around 1500 (over longer time periods there is also a slight upward trend due to increasing prices).

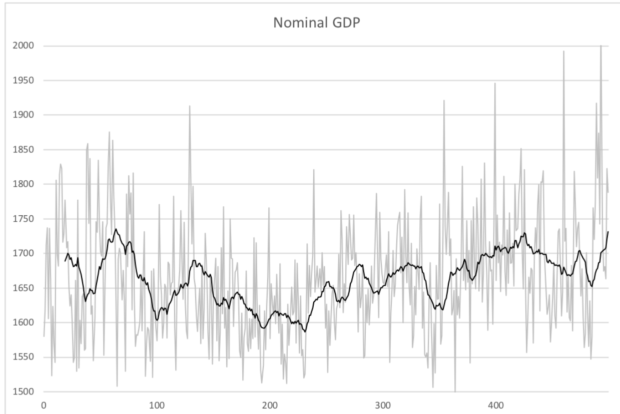

Figure 3.5: A sample simulation of the nominal GDP series under the benchmark calibration. The gray lines show the actual series and the black shows a 20 period moving average. The first 3000 periods of the simulation were dropped to avoid dependence on initial conditions. Therefore the figure shows periods 3000-3500.

Other variables show similar patterns. Figure 3.6 shows sample simulation results for consumption and investment (both in nominal terms). Once again we see a relatively stable mean with short term fluctuations around that mean. Comparing these two graphs with the graph of nominal GDP in Figure 3.5, we can see that investment is more volatile than GDP, which is more volatile than consumption. This pattern is confirmed by calculating the standard deviations and coefficients of variation for each variable. These statistics are described in Table 3.2. Prices also move pretty much in line with expectations. Consumer prices are relatively stable and persistent, while producer prices have much larger deviations from the mean (longer simulations demonstrate that the average price is still stable over long time periods). Figure 3.7 shows sample simulations for these price series. Again, the goal of the model is not to explicitly match real economic data at this point, but its ability to match general patterns regarding the volatility of aggregate variables is still encouraging.

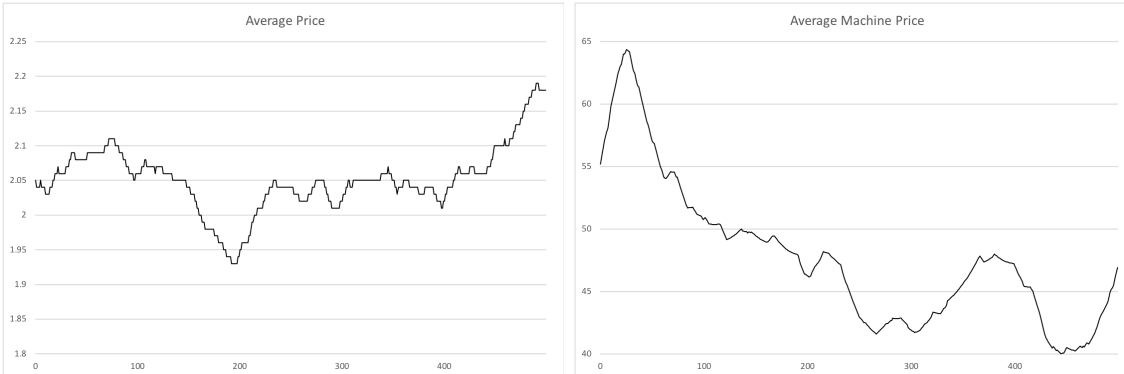

Figure 3.6: A sample simulation of consumer prices (left) and machine prices (right) under the benchmark calibration. The first 3000 periods of the simulation were dropped to avoid dependence on initial conditions. Therefore the figure shows periods 3000-3500.

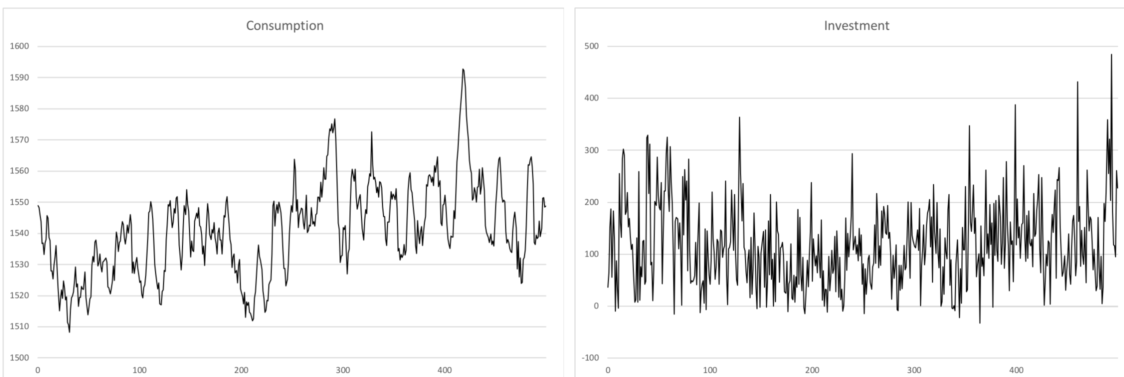

Figure 3.7: A sample simulation of the nominal consumption (left) and nominal investment (right) series under the benchmark calibration. The first 3000 periods of the simulation were dropped to avoid dependence on initial conditions. Therefore the figure shows periods 3000-3500.

The model does not do quite as well in the labor market. The specification for wage adjustments described in Section 3.2.2 is not nearly rich enough to enable realistic looking fluctuations in wages over time (in fact there is currently no way for nominal wages to fall at all). The path of wages is therefore not a useful variable in this setup. The unemployment rate (shown in Figure 3.8 is counterfactually high in level and excessively volatile. Section 3.5 will discuss some ways to deal with these issues in future iterations of the model.

Table 3.2: Sample Statistics

Variable	Mean	Standard Deviation	Coefficient of Variation
NGDP	1664.02	84.45	0.051
Consumption	1541.15	14.46	0.0094
Investment	117.49	80.43	0.685
Unemployment	22.62	1.71	0.075
Average Consumption Price	2.05	0.047	0.022
Average Machine Price	47.72	5.77	0.121

Figure 3.8: A sample simulation of the unemployment series under the benchmark calibration. The first 3000 periods of the simulation were dropped to avoid dependence on initial conditions. Therefore the figure shows periods 3000-3500.

Finally, the model can handle simple policy experiments. Policy is the area of the model that has the most room to grow. It should be possible to test various monetary and fiscal policies given the framework set out here, but doing so is beyond the scope of this paper. Here, I will simply provide a couple examples to show that changes in policy can have significant effects on aggregate variables.

The easiest type of policy to implement in the model as currently specified is through changes to the government's tax and benefit structure (laid out in detail in section 3.2.3). Figure 3.9 shows the average unemployment rate that the model produces under different levels of income tax (holding unemployment benefits fixed at the benchmark level) and different levels of unemployment rate benefits (holding income taxes fixed at the benchmark level). Recall that labor is supplied inelastically so there is no supply side effects from either policy. Instead, increased unemployment occurs due to lower aggregate demand from increased taxes or reduced benefits, which causes firms to cut back on production. Obviously this analysis is far too simple to be any use as a guide to real world policy, but this example simply serves to show that policy can have effects on the economy.

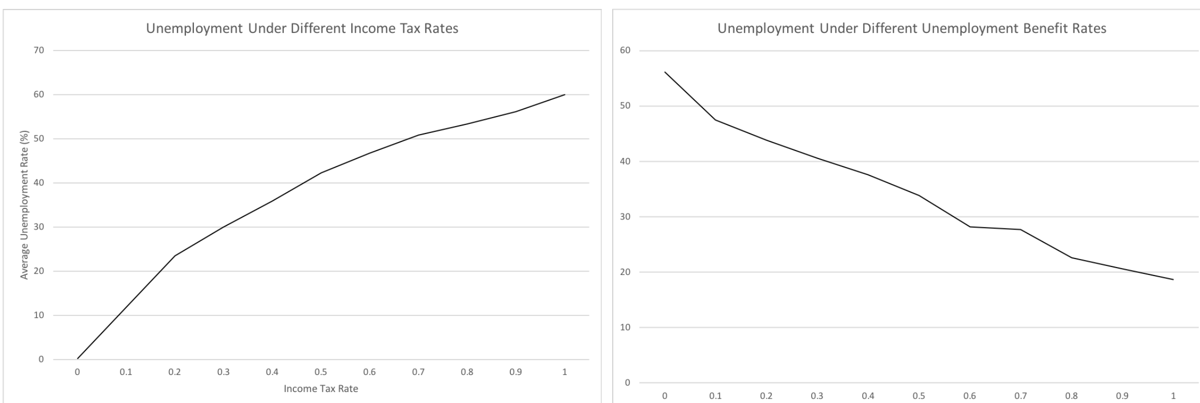

Figure 3.9: Left Panel: Average unemployment rates (vertical axis) under different levels of income taxes (horizontal axis). Right Panel: Average unemployment rates (vertical axis) under different levels of unemployment benefits (horizontal axis). Averages are over 1000 periods after dropping the first 500 periods of the simulation

The model can also incorporate various types of monetary policy. Again, more work needs to be done before experiments in the model can actually be applied to realistic situations, but it can still produce some intuitive results. In line with the expectations of standard models, increasing the money supply (remember that monetary policy works through helicopter drops to consumers) reduces unemployment but increases inflation. Figure 3.10 shows the change in the benchmark economy with an increasing money supply target (specifically with g_m set to 0.0005). As the figure shows, unemployment eventually drops to near zero with random spikes above and the volatility of nominal GDP increases drastically. Prices also rapidly increase as consumers try to spend an increasing amount of money on firms with a limited production capacity.

The model has some problems when it reaches full employment (zero percent unemployment) and inventories begin to approach zero, which causes the odd looking spikes in the nominal GDP graph. These issues will be discussed in the next section in more detail. Another policy experiment that avoids this issue is an unemployment target where the government increases the money supply only when unemployment falls below its target (note that this experiment is not currently possible in the standalone program - it requires a slight change in the code).

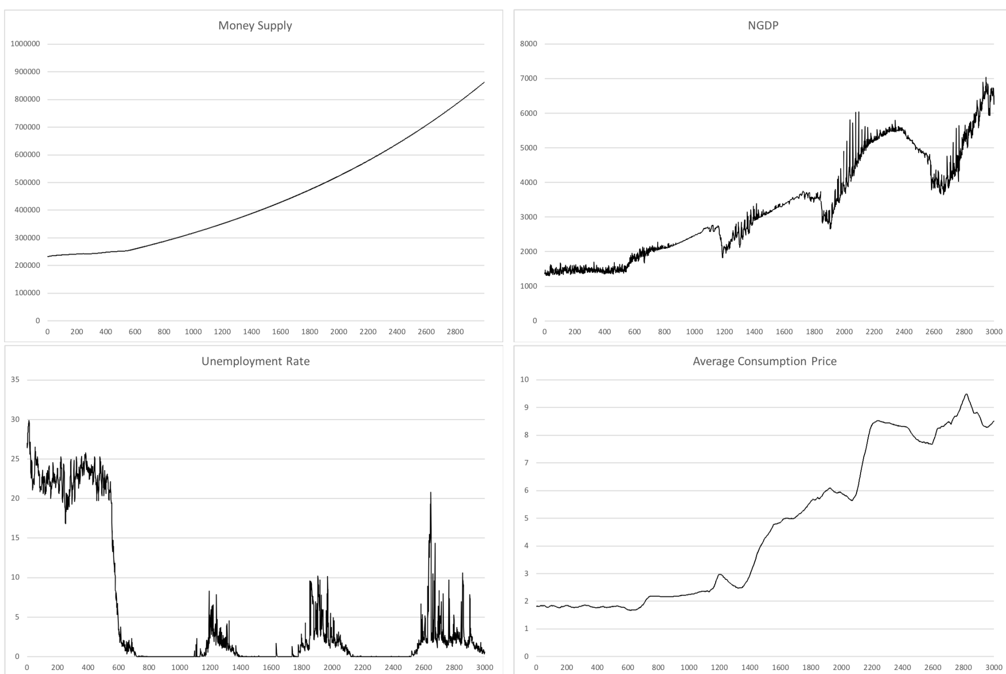

Figure 3.10: Simulation with a constantly increasing money supply target. From top left to bottom right: money supply, Nominal GDP, the unemployment rate, the average price of a consumption good. Simulation is over 3000 periods (500 periods dropped)

Figure 3.11 shows the results from a simulation with an unemployment target of 10%. This policy produces a much more stable economy than the constant money growth rate of the previous experiment. It accomplishes its goal of reducing unemployment relative to the benchmark case and produces much more stable paths for nominal GDP and the price level relative to the constant money growth case. Note that the money supply grows almost linearly in this case, compared to the exponential growth rate in the previous case.

These examples are meant to provide a basic intuition for how policy can work in the model, but they are by no means exhaustive. I expect that future work can examine much more interesting kinds of fiscal policies including government directly hiring or directly purchasing consumption and investment goods, as well as monetary policies like price level targeting, nominal GDP targeting, or a Taylor rule.

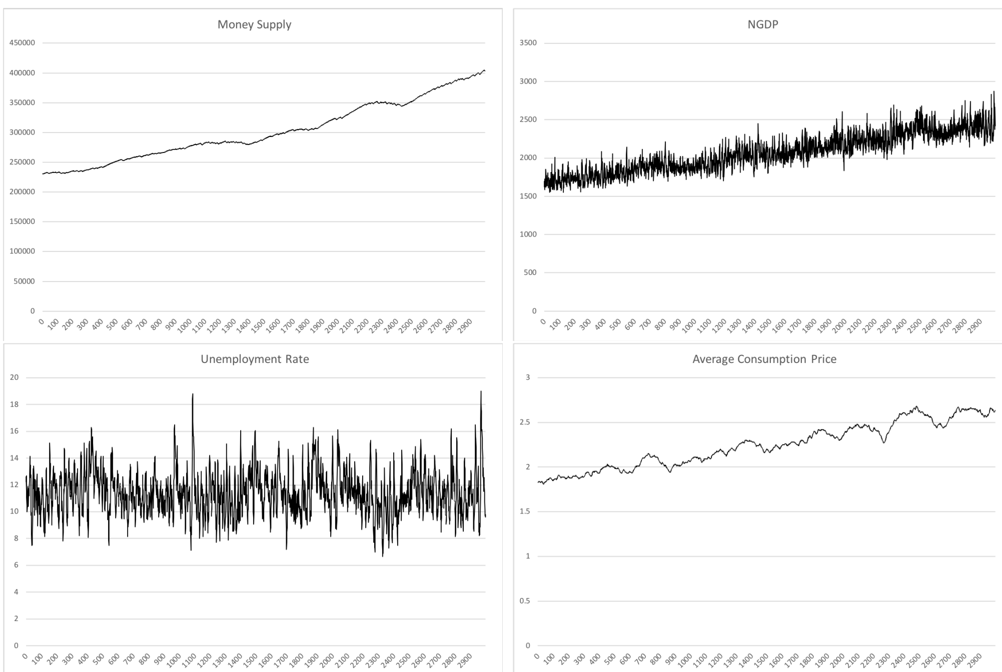

Figure 3.11: Simulation with an unemployment target. From top left to bottom right: money supply, Nominal GDP, the unemployment rate, the average price of a consumption good. Simulation is over 3000 periods (500 periods dropped)

3.5 Robustness/Extensions

Part of the inspiration for designing the model as an interactive one with the ability for users to change parameters is to make the model features more transparent. It is always a question in economic models whether the author simply cherry picked parameter values to make their model look good. Here, I allow the user to test their own parameter values to see how different variables can affect the results. To that end, in this section I offer some of the patterns I have observed in designing the model.

Starting with economy wide parameters, the economy is relatively robust to changes in the numbers of agents. Changing the number of consumers produces little qualitative difference in key variables unless the ratio of consumers to firms becomes too small. Without a large number of customers per consumption firm, the customer market setup breaks down and prices drop without firms gaining any customers, leading to an extremely low (and ultimately unprofitable) price level. The same logic applies to the number of consumption firms relative to investment firms. There must be sufficiently many consumption firms to allow price competition between investment firms to work properly. The other concern with the number of agents is computational. Too many agents causes the program to run slowly. These concerns led me to choose the benchmark calibration of 1500 consumers, 100 consumption firms (15 consumers per firm) and 5 investment firms (20 consumption firms per investment firms).

Consumer specific parameters are relatively unimportant to the working of the model. Changing mpc affects the economy in predictable ways and the excess cash spending rate can affect the persistence of consumption, but these are not especially interesting changes. The desired cash holding of consumers needs to be high enough that cash holdings do not consistently hit zero, but otherwise is basically unimportant.

One potential improvement to consumer behavior in the model is to allow marginal propensity to consume to adjust dynamically. Theoretically, adding dynamic savings could allow discussion of phenomena like the paradox of thrift where consumers attempt to increase savings only to see aggregate savings fall as demand collapses. One way to implement this idea is to use their current cash holdings to adjust their savings rate. If cash holdings are too low, consumers increase their rate of saving and decrease it when holdings are too high. I attempted to implement this setup in previous iterations of the model, but changes in mpc tended to be too drastic and led to even more severe swings in cash holdings. More thought will need to be given to adjustments in saving rates.

The specifications for firm gain and loss are designed to be robust to changes in the speed of gain and loss. Looking at equations 3.7 and 3.8, we can see that they are naturally designed to keep the number of searching customers in a stable range. As the number of searching consumers increases, it becomes easier for firms to attract new customers and vice versa. In precious iterations of the model without this feature, the number of searching consumers tended to either fall to zero (which causes problems for the customer market price adjustment mechanism) or increase to the entire population (which causes consumption to go to zero).

One particular aspect of the model worth discussing here is that consumers who are searching have zero consumption. I justify this assumption by interpreting zero consumption as consuming previously purchased goods within their own home (which is not counted as consumption in that period in aggregate accounts). Future work could explicitly account for this interpretation by separating the purchase of a good and its consumption. In other words, consumers could hold a stock of goods that they run down and need to replenish by purchasing more.

Perhaps the biggest issue with the mechanics of the model is the potential for firms to hit zero inventories. In the price adjustment mechanism, there is currently no way for low inventories to lead to price increases. I chose to avoid including inventories as a factor for price adjustment in order to isolate the customer market side of the model, but future work could certainly explore more complicated pricing schemes. As it stands, the main way that the model breaks down is when monetary policy causes large increases in aggregate demand and firms cannot produce fast enough to keep up. Since there is no way for productivity to increase in the current model, the only way for firms to respond to excess demand is by increasing prices. When the money supply grows rapidly, firms the pricing mechanism does not allow firms to increase prices fast enough to reduce demand sufficiently. This issue is what causes the spikes in NGDP seen in Figure 3.10.

The other major category of parameters is those controlling the characteristics of machines. The speed of depreciation affects the profitability of both consumption firms (which have to replenish machines more quickly with quicker depreciation) and investment firms (which get to sell more with quicker depreciation). Machine capacity has similar effects as lower capacity requires firms to purchase more machines for the same level of production. The time to build a machine affects investment firms profitability and employment because with longer periods, more workers need to be tied up in investment firms at any one time.

As mentioned briefly earlier, one motivation for this setup of machines is to add a Hayekian side to the economy where recessions can develop because of a mismatch between plans and actual outcomes. When investment takes time to build, firms can undertake projects that turn out to be unprofitable. Still, more work needs to be done before any serious comparison to Hayek's work can be made. With only one type of investment good, Hayek's idea of the lengthening of the structure of production is meaningless. Previous versions of the model attempted to include longer term investment projects ("factories"), but the decisions of both investment firms (in terms of how much to produce and price to charge) and consumption firms (in terms of how much of each type of investment to buy and when to switch supplier) become much more complex.

Another notable omission is the lack of any sort of bond market. Without a functioning market for debt, the model cannot capture many of the most important effects that occur in real economies like the effects of changes in interest rates, debt financing, and bankruptcy. Although debt implicitly has a place in the model through negative cash holdings of firms, this simplification cannot mimic real world patterns or decision-making related to debt. Early versions of the model attempted to include a more complete bond market, but doing so raises many questions. Who should supply bonds? Only governments or firms as well? What duration should bonds be offered at and how can these durations be decided? What happens when debts cannot be repaid? Because the model produces interesting results without these complications, I decided to leave them for future work.

One final point that could be included in future versions of the model is a better implementation of entry and exit. Although we can interpret a firm with zero customers as "exiting" the market, this interpretation isn't quite right because the firm retains its price and cash holding and still makes decisions about pricing that affect the market. A better (but much more complex) way of allowing entry and exit in the model would be an evolutionary system where each firm can have slight differences in the types of decisions it makes (for example some firms could change prices more frequently). Then, competition between firms would lead the model to choose the ones that perform the best and push out underperformers. With this kind of dynamics, the model would be less dependent on specific decision rules developed by the modeler, and could instead be used to figure out which kind of decision rules actually work best in practice. Again, this feature was attempted but ultimately proved to add too much complexity to be included in this version. Additionally, without a functioning bond market, it is difficult to decide when a firm would exit the market since in this version they can always borrow more money. Each of these additions require significantly more thought before they can be included.

3.6 Conclusion

Agent based models have enormous potential, but in order to reach that potential careful considerations must be given to how to best illustrate and understand the results generated by these models. One of the reasons equation based general equilibrium models have proven so effective in the history of macroeconomic methodology is their ability to produce clean, clear results through analytic solutions. Simulation focused agent based models can never hope to offer that level of clarity. Instead, we must consider other methods to make the results useful for researchers and policy makers.

This paper offers one possibility for conveying the results of an agent based model. It sets out a standalone program that can be downloaded and used to run experiments on a simple economy. By allowing users to see results of changes in real time, it makes it easy to ask and answer questions about how changes in the economic environment and policy can affect aggregate variables over time.

The model presented here is intentionally simplified and not meant to be used for serious policy analysis at this point. Instead, it provides a basic starting point which will hopefully be built upon in future work. The foundation of Keynesian inventory adjustments and price setting based on customer markets offers an intuitive and flexible framework that lends itself to a number of extensions including financial markets and evolutionary entry and exit.

3.7 Appendix I: Running the Program

The program can be downloaded from the author's website at http://chrissurro.com/programs-code/ by clicking the "program" link. Clicking this link will download a folder called "Keynesian Dynamics," which contains the application file (also called "Keynesian Dynamics"). Running the program will open the parameter selection screen as shown in figure 3.3.3.

Parameters can then be freely changed. Before starting the simulation, the enter key must be pressed to finish editing the parameter value. This step allows the program to check if a permissible value was entered and will not accept non-numbers as inputs. If the numeric value of the parameter lies outside a certain range (e.g. most parameters cannot be negative, probabilities must be between 0 and 1), it will automatically be corrected to an acceptable value.

Pressing "Start Simulation" opens the main screen as shown in figure 3.3.1. The economy will automatically begin running and statistics adjust in real time. Hovering over an agent will show the statistics for that agent on the right of the application window. Clicking the agent will lock those statistics in place until another agent is clicked. Clicking any statistic will open a graph of that variable that updates dynamically as the economy runs. Closing the simulation window allows the user to input new parameter values and start a new simulation from t=0.

3.8 Appendix II: Swift Code

A link to download the source code for the program can be found at http://chrissurro.com/programs-code/. The program was created using Apple's SpriteKit toolset for the Swift programming language, which means it requires an Apple computer, the program XCode, and a (free) developer account in order to run the code. This choice was made because SpriteKit offers an environment that makes animation and other visual elements relatively easy to implement. The additional package "Charts" (Gindi, 2018) is used to generate the dynamic graphs. To enable running the program, cocoapods must be installed to enable the use of the Charts package. Instructions for installing cocoapods can be found here: https://cocoapods.org. The Pod file necessary to install the Charts pod has already been included in the project folder. Therefore, installing only requires opening terminal, typing "sudo gem install cocoapods," changing the working directory to the folder containing the Xcode Project, and typing "pod install." From here, opening the file "Keynes ABM 2.xcworkspace" (note not .xcodeproj) should show the files that make up the program.

The code consists of 15 swift files. The file "AppDelegate" is automatically generated by Spritekit and left unchanged with the exception of one line of code that makes the program quit automatically when all windows are closed. The files "Bond" and "Factory" are unused in the current version of the program.

Global variables are defined in the file "Util.swift." Each variable that needs to be accessed by multiple agents in the economy is instantiated here with an initial value. Variables are defined as arrays indexed by t, so this setup essentially defines the value of global variables at period 0. For $t > 0$, additional values are added to the array in other files in the program.

The setup of the parameter selection screen is contained mostly in the file "StartScreen.swift" and also requires the use of the "Main.storyboard" file. Each parameter that can be changed is given a textbox which then sets the value of the specified parameter. Acceptable parameter ranges are also set here so that entries cannot include negative numbers or probabilities over 1. It is also important to note that the file "ViewController.swift" resets all global variables whenever a new simulation begins. This prevents variables keeping their values from the previous simulation.

For the main screen of the program, most tasks stem from the main file "MainScene.swift." This file begins by setting up the screen with the user specified number of consumers, consumption firms, investment firms, and a single government. Each type of agent is then defined in its own file. Each individual agent is an object with a specific set of characteristics. For example, a consumer has properties that correspond to its wage, mpc, consumption spending, etc. These variables can be seen at the start of each class. Note that the classes "CFirm.swift" and "IFirm.swift" both inherit from the parent class "Firm.swift," which enables them to use shared methods for hiring and firing workers without redefining in both classes.

Running the economy occurs through the “update” method in the main file. First, global variables are appended with an additional 0 (for flows) or the previous value (for stocks) using the transition method. It is important that this step occurs before any other so that these variables can be changed at the appropriate times. Each agent also has its own set of variables that must be appended at the beginning of the period. From there, the program follows the outline given in 3.2.

One important note for changing the variables displayed either for the entire economy or for individual agents is that two pieces of the code control these labels (both in the MainScene file). First, in the initial setup of the economy (within the didMove(to view) method), an array of strings is allocated. These will be the variables that are displayed in the label. Next, the actual variables that correspond to those strings need to be defined. This occurs in the update method and is defined separately for each type of agent as well as the whole economy (so there should be 5 sets of matching string and variable arrays total).

Finally, the program can also be run in “No Visual Mode,” which eschews the interactive and visual elements in favor of speed. No Visual Mode is activated by setting the property noVisualMode to true in the mainScene file. This feature is obviously only useful if the results can be stored somewhere, so there is also an option to save the results to a file by setting the property saveResults to true. Note that in order to save results, a location must be defined at the very end of the update method. Variables to save can be changed by changing the string array csvText in the property list of mainScene as well as the corresponding variable list at the end of the update method.

Other specifics of the code can be found in comments within the code files.

REFERENCES

Anderson, S. P., de Palma, A., and Thisse, J.-F. (1992). Discrete Choice Theory of Product Differentiation. The MIT Press, Cambridge, Massachusetts.

Argente, D. and Lee, M. (2019). Cost of Living Inequality during the Great Recession. SSRN Electronic Journal.

Ashraf, Q., Gershman, B., and Howitt, P. (2016). How inflation affects macroeconomic performance: An agent-based computational investigation. Macroeconomic Dynamics, 20(2):558–581.

Ashraf, Q., Gershman, B., and Howitt, P. (2017). Banks, market organization, and macroeconomic performance: An agent-based computational analysis. Journal of Economic Behavior and Organization, 135:143–180.

Athey, S. and Imbens, G. W. (2019). Machine Learning Methods Economists Should Know About — Stanford Graduate School of Business. Working Paper.

Barattieri, A., Basu, S., and Gottschalk, P. (2014). Some Evidence on the Importance of Sticky Wages. American Economic Journal: Macroeconomics, 6(1):70–101.

Bernhardsson, E. (2018). Annoy: Approximate Nearest Neighbors in C++/Python.

Berry, S., Levinsohn, J., and Pakes, A. (1995). Automobile Prices in Market Equilibrium. Econometrica, 63(4):841–890.

Blinder, A., Canetti, E. R., Lebow, D. E., and Rudd, J. B. (1998). Asking About Prices: A New Approach to Understanding Price Stickiness. Russell Sage Foundation.

Boskin, M. J., Dulberger, E. R., Gordon, R. J., Griliche, Z., and Jorgenson, D. W. (1997). The CPI Commission: Findings and Recommendations. The American Economic Review, 87(2).

Brock, W. A. and Hommes, C. H. (1997). A Rational Route to Randomness. Econometrica, 65(5):1059–1095.

Broda, C. and Weinstein, D. E. (2006). Globalization and the Gains from Variety. The Quarterly Journal of Economics, 121(2):541–585.

Broda, C. and Weinstein, D. E. (2010). Product creation and destruction: Evidence and price implications. American Economic Review, 100(3):691–723.

Compiani, G. (2018). Nonparametric Demand Estimation in Differentiated Products Markets. SSRN Electronic Journal, pages 1–70.

Dawid, H. and Delli Gatti, D. (2018). Agent-based macroeconomics. Handbook of Computational Economics, (02):1–94.

Diamond, P. A. (1971). A model of price adjustment. Journal of Economic Theory, 3:156–168.

Dixit, A. K. and Stiglitz, J. E. (1977). Monopolistic competition and optimum product diversity. The American Economic Review, 67(3):297–308.

Doms, M. and Dunne, T. (1998). Capital Adjustment Patterns in Manufacturing Plants. Review of Economic Dynamics, 1(2):409–429.

Dosi, G., Fagiolo, G., Napoletano, M., and Roventini, A. (2013). Income distribution, credit and fiscal policies in an agent-based Keynesian model. Journal of Economic Dynamics and Control, 37(8):1598–1625.

Dosi, G., Fagiolo, G., Napoletano, M., Roventini, A., and Treibich, T. (2015). Fiscal and monetary policies in complex evolving economies. Journal of Economic Dynamics and Control, 52:166–189.

Dosi, G., Fagiolo, G., and Roventini, A. (2006). An Evolutionary Model of Endogenous Business Cycles. Computational Economics, 27(1):3–34.

Dosi, G., Fagiolo, G., and Roventini, A. (2010). Schumpeter meeting Keynes: A policy-friendly model of endogenous growth and business cycles. Journal of Economic Dynamics and Control, 34(9):1748–1767.

Dosi, G., Napoletano, M., Roventini, A., and Treibich, T. (2017). Micro and macro policies in the Keynes+Schumpeter evolutionary models. Journal of Evolutionary Economics, 27(1):63–90.

Ertöz, L., Steinbach, M., and Kumar, V. (2003). Finding Clusters of Different Sizes, Shapes, and Densities in Noisy, High Dimensional Data. pages 47–58.

Fagiolo, G., Guerini, M., Lamperti, F., Moneta, A., and Roventini, A. (2017). Validation of agent-based models in economics and finance. LEM Working Paper Series, No. 2017(23).

Feenstra, R. C. (1994). New Product Varieties and the Measurement of International Prices. The American Economic Review, 84(1):157–177.

Floros, D., Liu, T., Pitsianis, N., and Sun, X. (2018). Sparse Dual of the Density Peaks Algorithm for Cluster Analysis of High-dimensional Data. 2018 IEEE High Performance Extreme Computing Conference, HPEC 2018.

Gindi, D. (2018). Charts.

Gourio, F. and Rudanko, L. (2014). Customer capital. Review of Economic Studies, 81(3):1102–1136.

Greene, W. H. and Hensher, D. A. (2003). A latent class model for discrete choice analysis: Contrasts with mixed logit. Transportation Research Part B: Methodological, 37(8):681–698.

Greenslade, J. V. and Parker, M. (2012). New insights into price-setting behaviour in the UK: Introduction and survey results. Economic Journal, 122(558):1–15.

Gualdi, S., Tarzia, M., Zamponi, F., and Bouchaud, J. P. (2015). Tipping points in macroeconomic agent-based models. Journal of Economic Dynamics and Control, 50:29–61.

Hayek, F. (1932). Prices and Production. MacMillan, New York.

Hottman, C. J., Redding, S. J., and Weinstein, D. E. (2016). Quantifying The Scources of Firm Heterogeneity. Quarterly Journal of Economics, 131(3):1291–1364.

Howitt, P. and Clower, R. (2000). The emergence of economic organization. Journal of Economic Behavior & Organization, 41(1):55–84.

Imbs, J. and Mejean, I. (2015). Elasticity optimism. American Economic Journal: Macroeconomics, 7(3):43–83.

Jaravel, X. (2019). The Unequal Gains from Product Innovations: Evidence from the US Retail Sector. The Quarterly Journal of Economics, 134(2):715–783.

Kydland, F. and Prescott, E. (1991). The econometrics of the general equilibrium approach to business cycles. The Scandinavian Journal of Economics, 93(2):161–178.

Mehmood, R., El-Ashram, S., Bie, R., Dawood, H., and Kos, A. (2017). Clustering by fast search & merge of local density peaks for gene expression microarray data. Scientific Reports, 7:1–7.

Nevo, A. (2011). Empirical Models of Consumer Behavior. Annual Review of Economics, 3(1):51–75.

Orcutt, G. H. (1950). The review of Economics and Statistics. MIT Press, 32(2):117–132.

Osharin, A. and Verbus, V. (2018). Heterogeneous Consumers and Trade Patterns in a Monopolistically Competitive Setting. SSRN Electronic Journal.

Paciello, L., Pozzi, A., and Trachter, N. (2016). Price Dynamics with Customer Markets. Working Paper.

Pandit, S. and Gupta, S. (2011). A Comparative Study on Distance Measuring Approaches for Clustering. International Journal of Research in Computer Science, 2(1):29–31.

Phelps, E. S. and Winter, S. G. (1970). Optimal Price Policy under Atomistic Competition.

Redding, S. and Weinstein, D. (2019). Measuring Aggregate Price Indexes with Taste Shocks: Theory and Evidence for CES Preferences. NBER Working Paper, pages 1–55.

Reinsdorf, M. E. A. (2007). How Should Inventory Investment be Measured in National Accounts? Working Paper, page 41.

Rodriguez, A. and Laio, A. (2014). Clustering by fast search and find of density peaks. Science, 344(6191):1492–1496.

Seppecher, P. (2012). Flexibility of wages and macroeconomic instability in an agent-based computational model with endogenous money. Macroeconomic Dynamics, 16(SUPPL. 2):284–297.

Seppecher, P. and Salle, I. (2015). Deleveraging crises and deep recessions: a behavioural approach. Applied Economics, 47(34-35):3771–3790.

Seppecher, P., Salle, I., and Lavoie, M. (2018). What drives markups ? Evolutionary pricing in an agent-based. Industrial and Corporate Change, pages 1–23.

Simon, H. A. (1956). Rational choice and the structure of the environment. Psychological Review, 63(2):129–138.

Tisue, S. and Wilensky, U. (2004). Netlogo: A simple environment for modeling complexity. Conference on Complex Systems, pages 1–10.

Tito, M. D. (2016). Welfare evaluation in a heterogeneous agent model: How representative is the CES representative consumer? Economics Letters, 143:99–102.

Train, K. E. (2009). Discrete Choice Models with Simulation. Cambridge University Press, New York, NY, second edi edition.

Windrum, P., Fagiolo, G., and Moneta, A. (2007). Empirical validation of agent-based models: Alternatives and prospects. Jasss, 10(2).